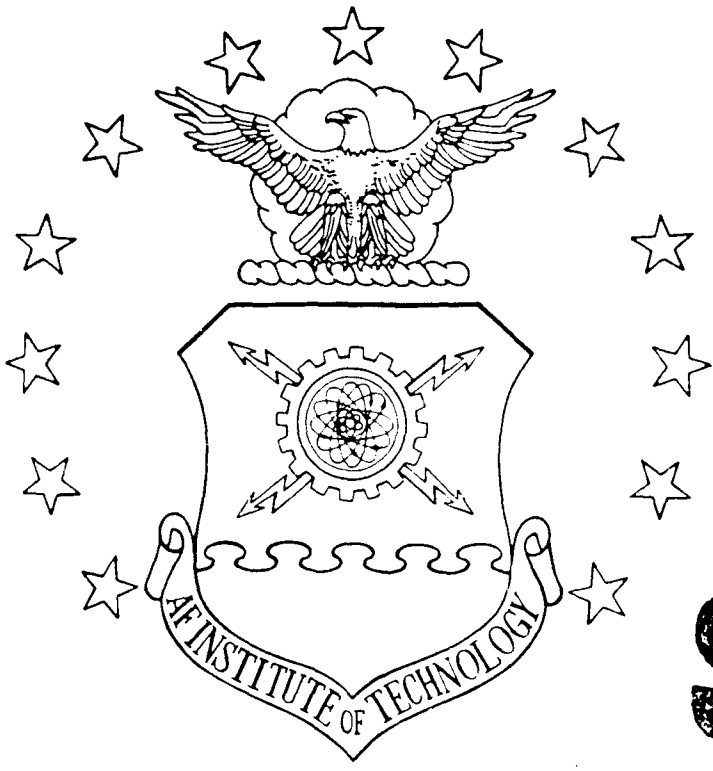
(2)



AN EXPERT SYSTEM FOR DEVELOPING A FULL SCALE DEVELOPMENT STATEMENT OF WORK

THESIS

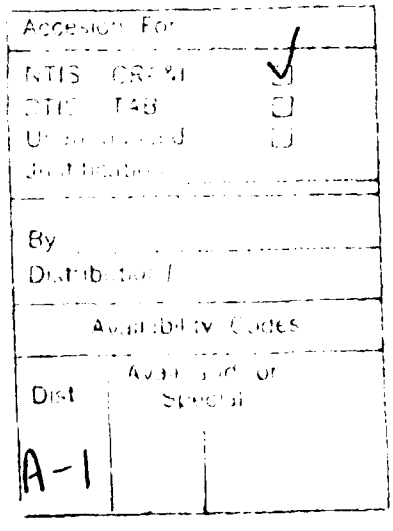
Keith A. Dierking
Captain, USAF

AFIT/GSM/LSM/89S-6

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The contents of the document are technically accurate, and no sensitive items, detrimental ideas, or deleterious information is contained therein. Furthermore, the views expressed in the document are those of the author and do not necessarily reflect the views of the School of Systems and Logistics, the Air University, the United States Air Force, or the Department of Defense.



AFIT/GSM/LSM/89S-6

AN EXPERT SYSTEM FOR DEVELOPING A FULL SCALE DEVELOPMENT STATEMENT OF WORK

THESIS

Presented to the Faculty of the School of Systems and Logistics
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Systems Management

Keith A. Dierking, B.S.
Captain, USAF

September 1989

Preface

The purpose of this study was to develop an expert system for developing a Full Scale Development Statement of Work. The methodology used in this study can be applied to develop contracts, item specifications, Program Management Plans (PMPs), Contract Data Requirement Lists (CDRLs), Test and Evaluation Maste Plans (TEMPs), and Acquisition Plans.

Programming was done with the help of the Knowledge Garden KnowledgePro System. KnowledgePro is a software package which combines the techniques of expert systems and hypertext. Additionally, I wish to thank the people at Knowledge Garden for their assistance over the phone in solving several problems which occurred in getting their KnowledgePro system to fit this research study.

In researching, writing and programming the software for this thesis, I had help from several people. I wish to thank Lieutenant Colonel Holt for his help in the later stages of the study and in developing the final product. I thank Captain Mun Kwon for his assistance during the early stages of this study. I thank the support from Rome Air Development Center at Griffiss AFB, New York, and Electronic Systems Division at Hanscom AFB, Massechusetts, for providing the majority of the rules which made this project possible. I thank Mr. Patrick Price for the time he took from his busy schedule in supporting this thesis study. Finally, I wish to

thank my wife, Kathy for her support and patience during the many hours I spent working on this research study.

Keith A. Dierking

Table of Contents

List of Figures

List of Programs

Abstract

The purpose of this project is to determine the need and feasibility of developing an expert system to assist in the development of a Statement of Work (SOW) for the Full Scale Development Phase (FSD) of the acquisition cycle. This project also determines the feasibility of transporting this expert system to a microcomputer.

The methodology involves a six step process for developing a small scale expert system. The first step involves choosing a tool. The tool chosen for this project is KnowledgePro by Knowledge Garden. The second step involves defining the problem Once the problem is defined, a determination is made on whether an expert system is appropriate for this project using a model developed in another thesis project. The third step involves developing the system. The fourth step involves developing a prototype system. The prototype system was developed on an IBM compatible computer using the KnowledgePro development system. The fifth step involved expanding, testing, and revising the system. The sixth step involved maintaining and updating the system. Beside the six steps for developing the expert system, the system was validated.

The project resulted in a microcomputer based expert system for assisting managers in the development of a FSD SOW.

The system produces a document in Work Breakdown Structure as called out by MIL-STD-245B consisting of essential SOW entries, potential documents to incorporate into the SOW, and action messages which provide direction in writing the SOW.

The benefits from this research is not limited to the development of a SOW. The methodology used in this research project can be used in the development of item specifications, contract development, Test and Evaluation Master Plans (TEMPs), acquisition plans, Program Management Plans (PMP's), and Contract Data Requirements Lists (CDRLs).

AN EXPERT SYSTEM FOR DEVELOPING A FULL SCALE DEVELOPMENT STATEMENT OF WORK

I. Introduction

The Department of Defense (DOD) purchases a wide variety of complex weapon systems, like aircraft, from industry. To ensure that industry builds a system which meets the governments requirements, DOD contracts include a document called the Statement of Work (SOW). (13:11) The purpose of the SOW is "to communicate the tasks and the work effort the government wants the contractor to perform. [1:10]."

Often SOWs do not achieve their main purpose of successful communication of the Government's requirements. The Aeronautical Systems Division (ASD) of Air Force Systems Command has expressed concern over the decline of the quality of Statements of Work citing "redundancy and ambiguities as frequent problems that surfaced in contract reviews [1:12]." Also, SOW developers may incorporate too few or too many requirements into the SOW. Due to the failure of SOWs to successfully communicate the Government's requirements, many projects have cost, schedule and performance problems.

An Air Force Institute of Technology (AFIT) research project was performed to determine what factors contribute to

successful SOW communication. The factors the research project looked at included the following: "requirements described by the SOW, the consistency of the SOW, the internal organization of the SOW document, cross-referencing of the SOW to other contract documents, SOW language clarity, the participation of the contractor in SOW preparation, and the effort of the Government team tasked with the development and writing of the SOW [1:40]." Of these factors, SOW language clarity and requirements description were found to provide the greatest contribution to successful SOW communication. (1:41)

The Electronic Systems Division at Hanscom Air Force Base in Massachusetts developed a computer software package to assist in the development of a SOW and other acquisition documents. This system is called the Computer Generated Acquisition Document System (CGADS). CGADS produces a draft of the desired document which must be tailored by the program manager for a specific program.

A survey given to students taking an AFIT professional continuing education course on Air Force project management, Systems 200, found several shortfalls with CGADS. First, CGADS is written for ESD contracts only. CGADS references ESD documents and organizations to produce the draft document. This makes adaptability by other product divisions difficult. Second, CGADS resides on a mainframe computer at ESD and a few other Air Force bases. This makes accessibility difficult for program managers, especially those without the proper equipment to connect directly to the mainframe computers. Third,

CGADS does not produce a SOW in the Work Breakdown Structure (WBS) format as described by MIL-HDBK-245B, Preparation of a Statement of Work (SOW), and AFSCR 800-6, Statement of Work. MIL-HDBK-245B and AFSCR 800-6 are further described in chapter II. Finally, CGADS references outdated documents resulting from frequent changes in acquisition policies and regulations.

Specific Problem

ESD is presently developing an expert system of CGADS. This expert system, like its predecessor, will be geared toward ESD contracts. This system will also be located on a mainframe computer at Hanscom AFB which will make accessibility very difficult for most product divisions. Also, this system will not produce a SOW in WBS format.

The purpose of this research project is to demonstrate the need and feasibility of developing an expert system to assist in the development of a SOW. This project will also determine the feasibility of transporting this expert system to a microcomputer.

Research Benefits

Development of a microcomputer based expert system for the SOW development task has the potential to better assist managers, especially new managers, to create a SOW by making expert knowledge readily available to many people at one time. By this system being on a microcomputer, managers will be able to access the data produced by the expert system easier than trying to access a mainframe computer located at another Air

Force base. This system should produce a document in WBS format as called out in MIL-HDBK-245B. This system should be easily adapted by product divisions to the way they do business and the ability to change referenced documents when the need arises. This system will assist the SOW developer in reducing redundancy, avoid ambiguities, avoid ommissions, and avoid incorporating too many requirements in the SOW. Also, this project will demonstrate the need and feasibility of developing such an expert system.

Research Questions

In order to validate ESD's development effort and transporting the expert system to a microcomputer, the following questions must be answered:

1. Is the development of a SOW a suitable problem for application of an expert system?
2. Can the rules needed to write a SOW be encoded using an expert system software program?
3. What is the user's subjective evaluation of the assistance provided by the expert system for writing a SOW?

Scope and Limitation of Research

As stated above CGADS was designed to assist program managers in the development of various contractual documents. This research will focus only on the SOW portion of CGADS. More specifically, this research project will focus on the Full Scale Development (FSD), Type III SOW. A description of a Type III SOW is provided in the literature review.

II. Literature Review

Overview

This chapter begins by providing background information on the content of Statements of Work (SOWs), problems with SOWs, and the Computer Generated Acquisition Document System (CGADS). This chapter also provides background information on expert systems, and the expert system development process.

Discussion

The Department of Defense (DOD) purchases a wide variety of complex weapon systems from industry. The document used to begin the acquisition process for these systems is the Request for Proposal (RFP). The RFP describes the acquisition conditions, weapon system capability objectives and how the proposal should be written. Potential contractors respond to the RFP with a proposal which describes how they will develop the weapon system. The proposal describes the management technical and cost conditions under which they are willing to perform to meet the Government's requirements.

One component of the RFP which provides the basic framework of the contractual task requirements is the Statement of Work (SOW). The purpose of the SOW is to communicate the tasks and the work effort the government wants the contractor to perform. From the requirements communicated in the SOW, the offerors determine their costs and develop a proposal on their system development approach. (13:11)

Air Force System Command (AFSC) currently uses MIL-HDBK-245B, Preparation of Statement of Work (SOW), and AFSCR 800-6, Statement of Work, as guides in writing a SOW. MIL-HDBK-245B was written "to provide guidance to the requiring activity to obtain conclusive contract Statement of Work (SOW) for application to any lifecycle phase of material acquisition [2:iii]." AFSCR 800-6 was developed by AFSC to be used in conjuction with MIL-HDBK-245B. AFSCR 800-6 controls "the preparation, format, and content of contract SOWs and serve as a management tool at the program, functional, and operating levels [8:1]."

MIL-HDBK-245B describes five different type of SOWs. These SOWs include:

Concept Exploration, Type I. The concept exploration phase is the initial phase in the acquisition cycle. The objective of this phase is to explore alternative design concepts. The technical requirements are usually in the form of goals or objectives. The Type I SOW is used to define these technical requirements in sufficient detail to provide guidance to the contractor. (2:7; 8:1)

Demonstration and Validation Phase, Type II. The objective of the Demonstration and Validation Phase is to analyze design alternatives and decide whether to go on to Full Scale Development. (2:7) The Type II SOW is used "when the work tasks are expressed objectively or as goal attainment [8:1]."

Full Scale Development, Type III. During this phase, the contractor will design and test a preproduction prototype system. (8:1) The Type III SOW is used

> concurrently with the specification to indicate the need for various system effectiveness program tasks, publications, training, integrated logistic support requirements, configuration management requirements, management systems, supply support tasks (provisioning), quality program requirements, metrology and contractor services. [2:8]

Production and Deployment Phase, Type IV. The objective of this phase is to produce the system designed in the previous phases and deploy them for operational use. The Type IV SOW state the tasks required to produce and control the design units. (2:9; 8:1)

Nonpersonal Service Contract, Type V. A Type V SOW is required when the need for contractor's services is required independent of the procurement contract. (8:2)

SOW Development Methods. Several methods are used in an attempt to develop a SOW which communicates the Government's requirements. These methods include using a previously written SOW, using policies and procedures like MIL-HDBK-245B, or by using the knowledge of someone who has written a SOW.

The Electronic Systems Division at Hanscom Air Force Base in Massachusetts saw a need for developing a computer software package to assist in the development of a SOW and other acquisition documents. This system is called the Computer Generated Acquisition Document System (CGADS). CGADS was developed to help the novice SOW writer, avoid redundancy,

prevent the ommission of requirements, and prevent the over specification of requirements.

Computer Generated Acquisition Document System. CGADS is a computer software program hosted on a main frame computer to assist program managers in the development of various contractual documents which successfully communicate the Government's requirements. CGADS produces a draft version of various documents. One of these documents is the Statement of Work. Other documents generated by CGADS includes a Reliability Centered Maintenance (RCM) Analyses Plan, Contract Data Requirements Lists (CDRLs), Multi-Year Affirmative Action Program (EEO/AAP) Mini Plans, Program Management Plans (PMPs), Acquisition Plans, and Test and Evaluation Master Plans (TEMPs).

CGADS obtains information from a user by asking a series of questions. Based upon the responses, CGADS outputs a draft version of the documents selected and a set of action messages to provide the user assistance in developing the document. A typical CGADS action message may look like the following:

** ACTION MESSAGE **
CDRL: DI-E-3120 AND DI-E-3119B NORMALLY APPLY. TAILOR AS REQUIRED.

The above action message tells the SOW developer that data items DI-E-3120 and DI-E-3119B should be incorporated into the CDRL. Also, these data items should be tailored according to the weapon system program requirements.

The draft document must be tailored by the program manager for a specific program. This task is performed by reviewing the action messages provided in the output, and the documents referenced by CGADS. The program manager may have to tailor the document according to other Air Force policies and regulations.

Expert Systems

Another method to assist in the development of a SOW is the use of expert systems. The use of expert systems for the development of a SOW is the focus of this research. Therefore, the remainder of this literature review focuses on expert systems.

Experts. In order to better understand what an expert system is, you must understand what an expert is. Experts are people who use their experience and knowledge to solve certain type of problems. (11:1) Paul E. Johnson, a behavioral scientist, describes an expert as follows:

> A person who, because of training and experience, is able to do things the rest of us cannot; experts are not only proficient but also smooth and efficient in the actions they take. Experts know a great many things and have tricks and caveats for applying what they know to problems and tasks. [17:5]

Experts are described as having the following characteristics:

1. Experts use their knowledge to solve problems.
2. Experts can explain their solutions.
3. Experts learn by experience.

4. Experts know their knowledge limitations.
5. Experts can solve problems quickly and efficiently.
6. Experts can restructure their knowledge.
7. Experts know when to break the rules.
8. Experts can determine relevant information. [5:16-17; 17:75]

In general, interaction with an expert involves exchange of information and knowledge. An expert is versatile and able to make a sensible judgment on how to best solve a problem. (5:16-17; 17:75)

Expert Systems. An expert system tries to emulate limited aspects of human thinking. An expert system can be defined as:

> An intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solution. Knowledge necessary to perform at such a level, plus the inference procedures used, can be thought of as a model of the best practitioners of the field. [3:5]

Expert systems are often considered a branch of artificial intelligence along with natural language processing and robotics. (4:3) Expert systems differ from most artificial intelligence programs in the following ways:

- expert systems deal with a narrowly defined problem
- it deals with realistic problems requiring human expertise
- it exhibits high performance in both speed and reliability
- in order to convince its users, the expert system must explain and justify its solutions. [9:1]

Conventional Programs	Expert Systems
Representation and use of data	Representation and use of knowledge
Knowledge and control integrated	Knowledge and control separated
Algorithmic Process	Heuristic Process
Effective manipulation of large data bases	Effective manipulation of large knowledge bases
Midrun explanation impossible	Midrun explanation desirable and achievable
Oriented toward numerical processing	Oriented toward symoblic processing
Maintained by programmers	Maintained by knowledge engineers and experts
Program sometimes difficult to modify	Knowledge base easy to modify

Figure I. Conventional Programs Versus Expert Systems (4:8)

Expert systems and conventional programs differ in a variety of ways. Some of these differences are listed seen in Figure I. Conventional programs often manipulate data bases. Expert systems manipulate knowledge bases. Conventional programs use algorithmic techniques to solve problems. Expert systems use heuristic techniques.

Another important difference between conventional programming and expert systems is how the systems are developed. In conventional programming the programmer meets with an expert for a short time to design a system. Once the system is designed, the programmer goes off and writes the computer program. The time period between when an expert initially meets with the programmer and the time the system is delivered can be extensive. Experts systems are developed by knowledge engineers in constant interaction with experts. These experts assist the knowledge engineer to identify the knowledge and develop inference strategies to be encoded into the expert system. (4:8-9)

Expert systems execute logical reasoning, similar to the way experts think. Thus, facts are linked with solutions to problems. (12:3) Expert systems use heuristics to arrive at a solution. Heuristics are rules of thumb and often called the 'art of good guessing'. (17:74) Heuristics enable

> experts, human or machine, to recognize promising approaches to problems, to break problems down into smaller components, to get around incomplete data, and to make educated guesses when necessary. [17:74]

Heuristics	Algorithms
If it ain't broke, don't fix it	1 + 2 = 3
The lower the price/earnings ratio the better buy a stock is likely to be.	The price/earnings ratio equals the stock's market price divided by the firm's latest retained earnings per share.
Select moves which protect the center of the chess board.	Black pawn at square D7, and square D6 and square D5 are empty, move pawn from D7 to D5.
If the car engine doesn't turn over when you turn the key, first check for loose wires, then for a dead battery.	For I = 1 to 10 X(I) = 2*10+I Next I
These are rules of thumb, proven generally reliable through experience, but not always correct. They are concepts, and can't be reduced to numbers.	These are exactly stated formulas which can be logically proven. Put in correct numerical data and you will get the correct answer.

Figure II. Contrasting Heuristics and Algorithms (14:29)

An example of heuristics is the following:

> If there is no protective circuit then the transistor will be destroyed by thermal runaway. [16:44]

A heuristic differs from an algorithms. (see Figure II) Algorithms are formulas which produces a correct answer when numerical data is inserted into that formula. Therefore, expert systems combine knowledge with heuristics to solve real world problems. Figure II contrasts heuristics with algorithms.

Knowledge. The process of building an expert system is called knowledge engineering. (6:23) The knowledge engineering team consists of a knowledge engineer, implementor, and an expert. The knowledge engineer assumes the roles of analyst and willing to perform to meet the Government's requirements. The third person of the team is the expert. Without his expertise the system can not be built. (16:23)

The main components of an expert system are the knowledge base, inference engine, and input/output interface. The knowledge base contains the experts knowledge about a particular task. The input/output interface allows the user to input information and allows the system to ask questions and state conclusions. (5:23). The inference engine "conducts the consultation with the user [4:49]." The inference engine performs the following tasks:

> Contains rules to tell how to solve a problem efficiently.
> Compares knowledge rules with facts about the problem.
> Tracks the status of the hypothesis being tested, and those that have been confirmed or rejected.

- Asks questions to obtain needed data.
- States conclusions to the user.
- Explains the chain of reasoning it used to reach a conclusion. [18:116]

Expert System Applications. Expert systems can be applied to several different general types of problem solving as shown in Figure III. Some types of problems require interpretating data, instructing students, or debugging a malfunction. Problems that require the use of heuristics and searching procedures to arrive at solutions are good candidates for expert systems. The development of an expert system requires an expert in the desired problem domain. This expert must be available throughout the development of the expert system. A good test to determine if a problem is suited for an expert system is the telephone test. If an expert can guide a novice through a problem over the phone then the task could be implemented by an expert system. (19:130)

Expert System Advantages. Expert systems offer some advantages over human experts. Experts can only be at one place at a time. Expert systems can be copied and made available to many more people to solve tough problems. Events frequently retire or leave the organization. But then, an expert's knowledge can be captured instead of lost. An expert system can also combine the knowledge of several experts to allow increased and more thorough processing.

Interpretation	Inferring situation descriptions from sensor data
Prediction	Inferring likely consequences of given situations
Diagnosis	Inferring system malfunctions from observables
Design	Configuring objects under constraints
Planning	Designing Actions
Monitoring	Comparing observations to plan vulnerabilities
Debugging	Prescribing remedies for malfunctions
Repair	Executing a plan to administer a prescribed remendy
Instruction	Diagnosing, debugging, and repairing student behavior
Control	Interpreting, predicting, repairing, and monitoring systems

Figure III. Generic Types of Problem Solving (4:94)

Expert systems have the capability to add new information easily to aid in the decision making process. (18:197)

Expert systems can operate consistently under stress and time critical situations. When under pressure human experts can miss important factors necessary in the decision making process. Expert systems can make these decisions without leaving out any important information. (18:84,197) Expert systems are consistent. They are

> available and fully functional for 24 hours a day, not to get tired, not to suffer from Monday morning blues or Friday afternoon impatience, and to give the same answer to beggars and kings. [16:16]

Another advantage of expert systems is in the training role. Turnover can require a constant need to train new people. Training can be a costly and time consuming process. Expert systems can assist in the training of personnel by showing correct choices among alternatives. They allow the experts to continue working in their area of expertise minimizing time for training. Expert systems can ask questions, provide solutions, and provide the logic behind the solution. If a flaw is found in the reasoning process, the flaw can easily be modified. (18:197)

Expert System Disadvantages. Expert systems also offer certain disadvantages. One of the disadvantages comes in the technical arena. Most expert systems require large computers to handle the data and process the data quickly enough to make the expert system useful. This can increase the cost and

reduce the availability of the expert system. Personal computers are becoming powerful enough and fast enough to handle significantly large expert systems. When using a personnal computer, the expert system is limited in size. (18:204) Another disadvantage involves conceptual problems. Expert systems deal with a large amount of data. A problem exists on deciding what data needs to be incorporated into the expert system and integrating the data in a format useable by the expert system. Also, the expert system developer must deal with multiple experts who frequently disagree on how to approach the same task. (18:204)

The development of an expert system requires experienced personnel. Often, a scarcity of programmers, trained users and operators, and other experts exists in a problem area. (18:204)

Development Process

The procedure to develop an expert system differs slightly depending on the size of the expert system. The development of a large scale expert system will be discussed and then the development of a small scale expert system.

Large Scale Expert System Development. Expert systems are developed in six independent phases:

Phase I: Selection of an appropriate domain
Phase II: Development of prototype system
Phase III: Development of a complete expert system
Phase IV: Evaluation of the system
Phase V: Integration of the system
Phase VI: Maintenance of the system. [4:196]

Phase I: Selection of an Appropriate Domain.

Selection of an appropriate problem requires several activities including:

- Identifying a problem domain and specific task
- Finding an expert willing to contribute expertise
- Identifying a tentative approach to the problem
- Analyzing the costs and benefits of the effort
- Preparing a specific development plan. [4:197]

Identifying a specific problem for expert system application is the most critical part of the system development. If an inappropriate problem is identified, the effort may run into design problems and high costs. (4:197)

The problem selection criteria can be divided into three different groups: system is possible, system is justified, and system is appropriate. (12:24)

Figure IV lists seven different characteristics that must be met for an expert system possible. One important characteristic is that an expert must exist. Without any experts, the system can not be developed. (4:197-201)

Figure V lists several problem domain characteristics any one of which could justify expert system development. Meeting any one of the characteristics is enough to justify the time and cost involved in the development of the expert system. (4:197-201)

Figure VI lists the characteristics which decide whether an expert system is appropriate for a particular problem. Note that the characteristics are broken up into three different factors: nature, complexity, and scope. The nature of the

Task does not require common sense

Task requires only cognitive skills

Experts can articulate their methods

Genuine experts exist

Experts agree on solutions

Task is not difficult

Task is not poorly understood

NOTE: All the above must be true in order to make expert system development possible.

Figure IV. Necessary Requirements For Expert Systems (7:25)

Task solution has a high payoff

Human expertise being lost

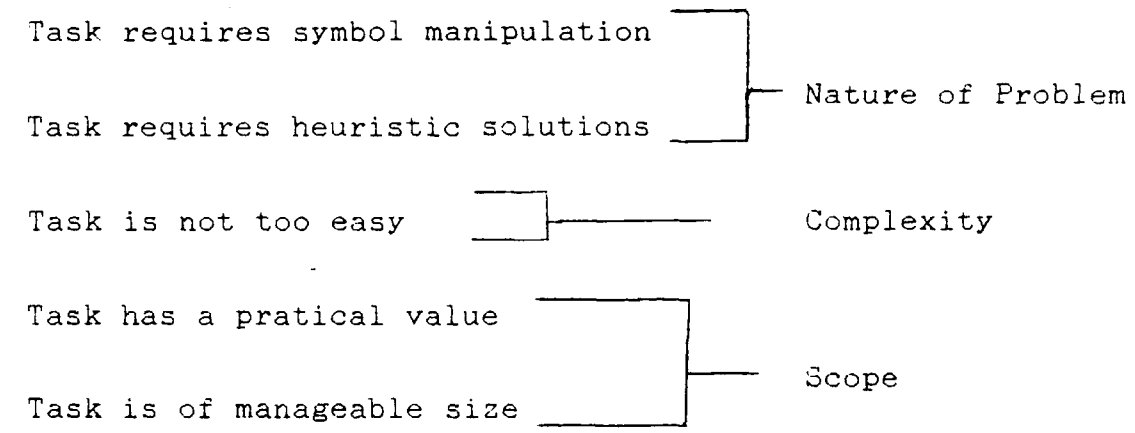
Human expertise scarce

Expertise needed in many locations

Expertise needed in hostile environment

NOTE: One or more of the above must be true in order to justify expert system development

Figure V. Justification for Expert System Development (7:27)



NOTE: All the above must be true in order to make expert system development appropriate.

Figure VI. Characteristics Appropriate to Expert System Development (7:29)

problem must require symbol manipulation and heuristics to bring about a solution. (4:197-201) The problem "must neither be too easy or too complex for a human expert [4:198]." The scope of the problem must "be sufficiently narrow to make the problem manageable and sufficiently broad to ensure that the problem has some practical interest [11:24]." The expert system should "satisfy a real need and be technically feasible [3:27]."

An important output of Phase I is a requirements document containing a list of potential expert systems, functions and knowledge base requirements. (3:27)

Phase II: Development of a Prototype System. The purpose of developing a prototype system is to "demonstrate the technical and economic feasibility of the intended expert system [3:28]." A successful demonstration will convince management to continue funding the project. Also, a prototype demonstration will lead the way for further knowledge base development. (3:28)

Development of the prototype requires several activities to take place. First, the knowledge engineer learns all he can about a particular problem domain and task. One method is for the knowledge engineer and expert to consider about four or five cases which represent the area of interest. The engineer studies the reasoning process used by the expert to solve the cases and develops problem solving strategies and heuristics. From this information, the knowledge engineer can

decide on an expert system tool and develop a prototype system. (4:201-203)

The knowledge engineer along with the expert will test the system to determine whether the strategies and heuristics used in representing the experts knowledge are adequate for a variety of cases. Once the system is functioning adequately, the knowledge engineer develops a detailed design plan for the complete system. (4:201-203)

Phase III: Development of a Complete Expert System. The prototype system and the detailed plan help the knowledge engineer develop the complete system. The knowledge engineer rethinks the design to determine if the heuristics used are adequate or if a modification is necessary. Additional heuristics are added to deal with the subtler aspects of a certain problem. Finally, the user interface is tailored to make it easy and natural for the user interface with the system. (4:203-205; 3:28)

Phase IV: Evaluation of the System. When the knowledge engineer and expert are satisfied that the system is complete, they will test the system against the requirements developed in Phase II. Also, other experts will test the system against a rigorous variety of cases to validate its decission logic. (4:205)

Phase V: Integration of the System. When the expert system is ready, the system is integrated into the work environment. This may involve making minor changes to the system to allow it to work in a particular environment and/or

interfacing the system with other data bases, instruments, and hardware to enhance its capability. Also, the knowledge engineer trains the users on how to maintain and use the system before withdrawing from the project. (4:205-206)

Phase VI: Maintenance of the System. As the knowledge to perform a particular task changes so must the expert system change. If the expert system was designed properly, the person in charge of the system can make the routine updates to the expert system without the knowledge engineer. (4:206)

Small-Scale Knowledge System Development. The discussion thus far dealt with the development of large scale knowledge systems. Harmon and King in their book "Expert Systems: Artificial Intelligence in Business" describe the development of a small knowledge systems in a different manner. Small knowledge systems can usually be developed without a knowledge engineers and the expert is frequently the user. An example would be a system to help clerks classify insurance applications. This system would be developed from the knowledge possessed by several experienced insurance application examiners. The examiners could enter their knowledge in an expert system shell. They would in turn use the system to make better decisions. (4:177)

Harmon and Key claim that small knowledge systems can be used and developed by middle management.

> Our own view is that small knowledge system building tools can and will be used by middle managers and training developers to solve a vast array of small, irksome problems. The individuals using these tools will not be "knowledge engineers" but will, instead, be people who are close to the problems. Senior application examiners will develop small knowledge systems that will provide assistance to new clerical personnel. Moreover, these same individual and their managers will also maintain and update the systems. [4:178]

The development of a small scale knowledge system requires the following steps:

1. Select a tool and implicitly commit yourself to a particular consultation paradigm.
2. Identify a problem and then analyze the knowledge to be included in the system.
3. Design the system. Initially this involves describing the system on paper. It typically involves making flow diagrams and matrices and drafting a few rules.
4. Develop a prototype of the system using the tool. This involves actually creating the knowledge base and testing it by running a number of consultations.
5. Expand, test, and revise the system until it does what you want it to do.
6. Maintain and update the system as needed. [4:178]

Note that the steps for developing a small scale expert system is similiar to the process of developing a large scale expert system. But, in the development of a small scale expert system the knowledge engineer's job has been replaced by the expert and the chosen tool. Therefore, knowledge systems will be created by the people who actually use them as in the examiner example previously mentioned. (4:194)

Harmon and King believe that small expert systems will show up in many business operations in the same way electronic spreadsheets were welcomed. Small experts systems will be welcomed because

> they will provide for vastly improved decision making and productivity in hundreds of small ways, and the result will be a revolution in the way business people think about knowledge, training, documentation, and procedural flows. [4:194]

Examples

MYCIN. Mycin is an expert system designed at Stanford University in 1972. Mycin was designed to assist medical personnel in the diagnosis of bacterial infections and prescribe the proper antibiotic to treat the infection. Mycin was designed to assist in four decisions: "does the patient suffer from bacterial infection, what organism is responsible, which drugs may be appropriate, and which of these to administer [12:8]." The rules used came from specialists in the field of bacterial infection. In a series of tests, Mycin performed as well as the specialists and significantly better than the nonspecialists. (16:8-9)

XCON. XCON was developed at Carnegie-Mellon University at the request of Digital Equipment Corporation (DEC). DEC wanted a system to assist their customers in deciding what items of equipment should make up a particular installation site. Installing a new system required the knowledge of a wide range of equipment available and limiting constraints at

the installation sites. DEC found that traditional methods did not work. So, they called on Carnegie-Mellon to develop a VAX based expert system. DEC calculated that without the expert system 80 staff positions would be required to perform the same tasks performed by XCON. DEC employed XCON to help the sales force in meeting the customers' needs. Also, XCON was used to assist in site preparation and in factory scheduling, and material control. (16:9-10)

Summary

The purpose of this chapter was to review the literature to provide a brief description of the SOW and provide a broad understanding of expert systems. A brief description of five different SOWs and on how they fit in the acquisition cycle was provided. The remainder of the review discussed expert systems along with their advantages and disadvantages.

III. Methodology

Overview

One method of developing a Statement of Work (SOW) is through the use of an expert system. Since only the Full Scale Development (FSD) SOW is the focus of attention for this project, the small scale expert system was chosen. The small scale expert system development process, as discussed in the literature review, was used to answer the following research questions:

1. Is the development of a Statement of Work (SOW) a suitable problem for application of an expert system?
2. Can the rules needed to write a SOW be encoded using an expert system shell?
3. What is the user's subjective evaluation of the assistance provided by the expert system for writing a SOW?

Step 1: Selecting a Tool

The targeted microcomputer for this project is an IBM compatible computer. Several software packages which can run on an IBM compatible machine were considered including those in the fields of word processing, database management, programming languages, and expert systems. The main criteria used in selecting a software package was the ease in updating the software program and text.

Step 2: Identify a Problem

The model used to answer the above questions was developed by Capt. Hazen in his AFIT thesis titled "How Can Air Force Civil Engineers use expert systems" and is found in Appendix A. In order for an expert system to be appropriate the task must meet the selection criteria of the model. Each of the five areas have a series of Yes/No questions. In areas I, III, IV, and V all the questions must be answered Yes in order for the task to meet that particular criteria. Area II only requires a Yes response to one of the four questions in that area.

Hazen's model was used to determine if an expert system is appropriate for developing a Statement of Work and therefore convert the present CGADS system into an expert system. The method chosen to answer the questions posed by the model is the interview process.

Interviewing was chosen over survey to obtain the information mentioned above. Interviewing allowed interaction with the expert and the ability to clarify any problems the expert had in understanding the questions firsthand. Also, interviewing allowed the ability to obtain more indepth information based on the expert's response to the questions.

Interviewing has several drawbacks. Interviewing involves taking up a good block of the expert's time as opposed to a survey which can be done more quickly and at a more convenient time for the expert. Assuming cooperation by

the expert, the benefits from interviewing outweighs the drawbacks.

The person interviewed is considered an expert in writing a SOW by his peers and through his experience.

Step 3: Design the System

The present CGADS system is designed to develop several documents, as discussed in a previous chapter, one of which is a Statement of Work. CGADS can assist in developing a Statement of Work for each phase of the contract. CGADS breaks down the FSD SOW development into several areas (e.g. engineering, program management, etc.). Each area is then broken down into several tasks (e.g. engineering is broken down into system engineering, human factors, value engineering, etc.). Each task consists of several YES-NO questions developed by experts at Electronic System Division (ESD) in that particular task area. The output depends on the responses to the YES-NO questions.

This thesis project uses these questions as a starting point in the development of an expert system version of CGADS. Some of the YES-NO questions were retained, while others were combined into one multiple choice question. The rules to handle the response to each question were obtained by attending a Design Review on the expert system version of CGADS being developed on a mainframe computer by ESD. Also, rules were developed by studying the operation of the present CGADS system. These rules determine what text to output and

which question to ask next based on the response to a particular question. The expert system also incorporates inferencing. Inferencing provides the ability to automatically answer other questions based on the response to a previous question. This project also redesigned CGADS so that it outputs the SOW in a Work Breakdown Structure (WBS) format as required in MIL-STD-245B. Figure VII contains the WBS structure used for this project.

Step 4: Develop the Prototype System

The system design above in step 3 will be reformated into a form understandable by the expert system shell KnowledgePro. KnowledgePro was selected because of the ease of programming in the higher order language it uses. Also, KnowledgePro lacks any significant bugs in its code which would prohibit use of its total capabilities as experienced with other expert system shells.

The expert system must also be redesigned to handle the limitations of the computer hardware. The system is designed to operate on an IBM-PC or compatible. The main limitation is main memory. The main memory is not large enough to handle the expert system shell and the entire knowledge base at one time.

The text required to build the output document is kept in ASCII files. The required text is taken from the ASCII file through the use of keywords. This text is both displayed in a window on the terminal screen and written out to a file. The

1.0 Scope

2.0 Listing of Data Items

3.0* Requirements

3.1 Hardware

3.2* Training

3.3* Peculiar Support Equipment

3.4* System Test and Evaluation
3.4.1 Development Test and Evaluation
3.4.2 Operational Test and Evaluation
3.4.3 Markups
3.4.4* Test and Evaluation Support
Preoperational Maintenance:
Preoperational Supply Support:

3.5* System/Project Management
3.5.1* Systems Management

3.5.1.1* Design Engineering
Human Factors:
Value Engineering:
Parts Control Program:
Electromagnetic Compatability:
Survivability/Vulnerability:

3.5.1.2* Logistic Engineering
Availability:
Maintainability:
Reliability:
Logistics Support Analysis:
Integrated Logistics Support:
Transportability:

3.5.1.3* Specialty Engineering
System Safety:
Aerospace Meteorlogical Environment:
Preservation, Packaging, and Packing:
Transportation:

Figure VII. Work Breakdown Structure For a Full Scale Development Statement of Work

3.5.1.4* Manufacturing Engineering
Quality Assurance:

3.5.1.5* Security
General Security:
Communications Security/Tempest:

3.5.1.6* Communications
Communications Long Lines:
Radio Frequency Management:

3.5.2* Project Management
3.5.2.1* Contract Work Breakdown Structure
3.5.2.2* Cost Information Systems
3.5.2.3* Cost/Schedule Control Systems
3.5.2.4* Schedule Management
3.5.2.5* Configuration Management
3.5.2.6* Data Management
3.5.2.7* Nomenclature
3.5.2.8* Manufacturing Management
3.5.2.9* Computer Resources Management
3.5.2.10* Travel

3.6* Data
3.6.1* Technical Publications
3.6.2* Engineering Data
3.6.3* Management Data

3.7* Operations/Site Activation
3.7.1* Real Property Facilities

3.8* Common Support Equipment
3.8.1 Organizational
3.8.2 Intermediate
3.8.3 Depot

3.9 Industrial Facilities
3.9.1 Construction/Conversition/Expansion
3.9.2 Equipment Acquisition or Modernization
3.9.3 Maintenance

3.10* Initial Spares and Repair Parts

* This paragraph is produced by the expert system developed for this project.

Figure VII. (concluded)

completed output is in a ASCII file and can be edited by any word processing package.

Completion of this step resulted in the development of the initial expert system prototype.

Step 5: Expand, Test, and Revise the System

Once a prototype was developed, the system was presented to several experts, both familiar and unfamiliar with the present CGADS system, to obtain their comments. The experts were asked to provide recommended changes in terminology to make the questions and the text clearer. The experts were asked to recommend changes to make the interactive consultation flow in a more logical progression. The comments from the experts were incorportated into the final expert system.

The final expert system was compared against the present CGADS system to check for faulty wording or advice given in the logical flow and output, the differences were noted and incorporated into the final system.

Step 6: Maintain and Update System

Maintaining and updating the system must continue throughout the lifetime of the system. Maintaining and updating the knowledge base of the expert system requires the use of the KnowledgePro Development System.

Updating and Maintaining the expert system requires very little programming skills. As stated earlier, KnowledgePro uses a higher order language. This higher order language

allows you to write powerful programs with only a few meaningful commands. KnowledgePro uses Topics as its base. For this project, each topic consists of a question and rules to handle the response to the question. The rules consist of IF..THEN..ELSE statements which determines the block of text to output and the logical flow of the program from topic to topic. Therefore, by understanding the syntax of a few simple commands and studying the source code of the expert system, the expert system can be maintained. In contrast, the present CGADS system requires a thorough knowledge of FORTRAN and the availability of a mainframe computer.

The expert system can be maintained and updated by loading the source code into a word processor and saving the necessary changes. The source code must then be compiled by the compiler available in the KnowledgePro development system before it can be executed.

The KnowledgePro software package consists of a development system and a run-time system. The run-time system consists of everything offered by the development system except a compiler and can be freely distributed with the expert system package.

The text for the expert system is kept in a separate ASCII file. Any updates to the text can be made with any MS-DOS word processor. In contrast, the mainframe version of CGADS has the text either compiled along with the code or in a database. Therfore, this requires accessability to the

compiler and database system and full understanding of these systems.

Validation

The validation of the system consisted of a subjective evaluation by experts and nonexperts and comparing the system with the existing CGADs system. They each created a SOW by running several interactive consultations with the expertsystem. Their opinions on the usefulness of the system was used to evaluate its worth.

The second part of the validation process consisted of comparing the output of the expert system with the output of the existing CGADS system. The expert system was executed several times and an output was obtained for various combination of responses. The same combinations were tested on the existing CGADS system. Any differences between the operation of the two systems and their output were noted.

IV. Findings and Results

In chapter I, three research questions were identified that are considered important in successfully performing this research. This research was performed as described in chapter III. The research undertaken answered the questions in the following manner.

Problem Evaluation

Question 1. Is the development of a Statement of Work (SOW) a suitable problem for application of an expert system?

As discussed in Chapter 3, Capt. Hazen developed a model to determine whether an expert system is appropriate for a particular problem. This model consists of twenty-three questions as shown in Appendix A.

A civilian engineer, considered an expert in developing an expert system by his peers, was selected to be interviewed. The questions used in the interview are those mentioned above. The expert answered YES to all the questions except questions 2 and 6. The expert also responded with the following comments for the indicated questions:

1. Will the expert system tackle problems managers have tackled before?

YES. Alot depends on the application. Some SOWs are written for new system. The lessons learned are all you have to fall back on. (15)

8. Do the experts agree on solutions?

Experts generally agree on a boilerplate solution for developing a SOW. Since every program is different, the boilerplate solution would have to be changed to fit the application. (15)

9 Are experts better than novices at performing the task?

In most cases experts are better than novices in writing a Statement of Work. But, sometimes experts have a preconceived notion on the correct way of writing the SOW without looking at the true facts. (15)

21. Does the task require commonsense?

Overall, yes. Experts can use common sense through experience. But novices may be unable to apply common sense due to the lack of experience and rely solely on a particular model either on a computer or on hard copies. (15)

Since a YES response can be indicated for the five model areas of Realistic, Justified, Expertise, Task, and Other Considerations, the development of an expert system can be recommended for writing a Statement of Work.

Expert System

Question 2. Can the rules needed to write a SOW be encoded using an expert system shell?

As stated in chapter III, several software packages were examined. These software packages included wordprocessors, database software, programming languages, and expert system shells.

Word processing software was rejected due the difficulty the users will have in rearranging a large volume of text to produce the desired document. This rearranging of the text will require the knowledge of what text goes with certain conditions. Therefore, the system will be of no use to the first time developer.

Database management software was viable option for this thesis. A set of records can be developed with each record containing the appropriate text for a given condition. A program could be written to output text in the appropriate record for when a certain condition is met. This option was rejected because an updating of the program will require programming experience with the database. Also, decision making and inferencing rules required some extensive programming.

A program written in a programming language such as FORTRAN or PASCAL was another possibility. This option was rejected since extensive programming skills are required to perform updates to the program. Also, the output text would have to be built into the code. Therefore, the appropriate programming language compiler must be available to update the text.

The final software package and the one chosen for this project is an expert system. Experts systems are discussed in chapter II of this document. The main advantage an expert system has over the other packages discussed is that it allows programming using a higher level language than PASCAL or

FORTRAN. This allows easy updates by the "nonprogrammer" since extensive programming knowledge is not required. Most of the program requires the use of IF..THEN..ELSE statements. Also, an expert system allows the text required for this project to be keep either in a database or a standard ASCII text file. This allows updates to be performed to the text without the use of the expert system package.

Several expert system shells were considered for this research project. These shells included VP-EXPERT, GURU, and KnowledgePro. The original development effort started with VP-EXPERT. VP-EXPERT was later found unuseful due to bugs in the software package which prevented the use of several required commands. GURU was considered but not used beccause of its complexity and difficulty of use. KnowledgePro was finally selected as the expert system shell for this research project.

Another consideration was to either integrate the expert system with dBASE III+ or use a form of hypertext and keep the text in a seperate ASCII file. dBASE III+ has a field called a MEMO field which allows the capability to store large amount of text with each record. Therefore, the text required for this project could be seperated into records and kept in the MEMO field of the record. This will allow ease of maintenance and the use of a popular database software package. A problem arose in trying to extract data from the MEMO field. The expert system shells examined for this project were able to extract data from any field of the record except the MEMO

field. Therefore, the idea of integrating the expert system with a database software package was abandoned and the use of hypertext was examined.

The hypertext capability of KnowledgePro proved to be the best route to take. Hypertext allows the capability to extract the required data from a seperate ASCII file. This then enables the user to update this text with the use of any word processor and without the need to compile the text along with the main program code. Once the calling program extracts the required block of text, the text is displayed on the video screen and written to a ASCII file.

KnowledgePro performs the task of extracting text from an ASCII file through the use of the READ and FIND commands. The FIND command was chosen over the READ command due to problems caused by the READ command. The READ command tends to put End of File (EOF) marker(s) either at the beginning of the block of text being brought into the program or at the end of the file. This usually occurs if the text was orignially produced on an editor other than the KnowledgePro editor. This does not pose any problems when displaying the text, but prevents any text from being appended after that block of text. The FIND command, found in versions 1.4 or later, does not present the same problems as the READ command. A more detailed description of the system operation can be found in the Program Documentation, Appendix B.

As mentioned in Chapter III, the main memory is not large enough to handle the expert system shell and a large knowledge

base. The memory problem was solved by breaking the program up into separate independent modules. Each module was design to ask the questions necessary to produce the following main paragraphs: 3.2 Training, 3.4 System Test and Evaluation, 3.5 System/Project Management, 3.6 Data, 3.7 Operation/Site Activation, 3.8 Common Support Equipment, and 3.10 Initial Spares and Repair Parts. Modules were designed for subparagraphs: 3.5.1 Systems Engineering, 3.5.1.2 Logistics Engineering, 3.5.1.3 Logistics Engineering, 3.5.1.3 Specialty Engineering, 3.5.1.4 Manufacturing Engineering, 3.5.1.5 Security, 3.5.1.6 Communications, and 3.5.2 Project Management. Each module is executed via a menu selection. Once the module has completed executing, it is removed from memory. Any communication between modules is performed by writing to and reading from data files.

Validation

Question 3. What is the user's subjective evaluation of the assistance provided by the expert system in writing a SOW?

The system was demonstrated before four managers at a Systems Program Office. Two of the managers are considered experts by the their peers in writing a Statement of Work and the remaining two managers have either no experience or very little experience in writing a SOW.

All the managers interviewed were excited about the system. They felt that the system would be useful to both experts and novices. They felt that the output both provides

words for a generic SOW and a checklist to follow especially in determining which documents are appropriate for a particular SOW application. They liked the format of the output which combines essential SOW entries, action messages, and potential documents to include in the SOW.

Currently, a SOW from a present program is frequently used as a guide in writing a SOW for another program. The SOW writer will often include paragraphs from the old SOW without knowing why the text was included in the first place. Therefore, the SOW under development will include unnecessary information which may result in the over specification of the work to be performed and additional costs to the government. The managers interviewed felt that the expert system would help the SOW developer understand why a document should be included in the SOW or why a certain task must be included in the SOW.

Another feature the managers liked was the ability to modify the output text or the program source text found in the text files. They showed concern especially over the ability to update the text files. They felt that the text would need constant updating when document revisions change and the text needs to be modified to the way they do business in their System Program Office. When they realized this text is kept seperate from the compiled knowledge base and that the ASCII file can be updated by any word processor, they were sold on the system. When the managers examined the source code and the capabilities of the KnowledgePro programming language,

they became interested in obtaining a copy of KnowledgePro so they could modify the code for their particular use.

Finally, they liked the ability to use the system on a personnal computer instead of the inconvenience of working on a mainframe computer in another building.

They did mention a couple of drawbacks concerning the system. First, the system was written mainly for ESD type contracts and does not completely adapt to the way their System Program Office (SPO) operates. Secondly, with the system involves constantly updating the system. They felt that in order for the system to be useful someone must constantly update any information in the text especially the documents called out by the text.

Other Findings

The final expert system was compared to the present CGADS system. The expert system version generally produced the same data in the output but differed in the format of the output (see Figures VIII and IX). The output from the mainframe version is in a logical sequence with a seperate listing for essential entries and action messages. The PC version has the output in the order the questions were asked and requires some work on the word processor before a clean document is produced. The PC version has action messages and essential entries in one document along with the response to the particular question which produced a particular set of output. Also, note that the output of the expert system is in WBS

3.5.1.6 Communications:

COMMUNICATIONS LONG LINE:

// Long_Line_Communications-YES

** DOCUMENT CALL **

DCAC Circular 310-130-1	Submission of Telecommunications
Apr 76	Service Requests
	ALL

** ESSENTIAL ENTRY **
Requests for leased long lines to support this system must be submitted to the procuring activity not later than 150 days prior to the need date.

** ACTION MESSAGE **
SOW: COORDINATE WITH SYSTEMS ENGINEERING TO ENSURE THAT THIS REQUIREMENT HAS NOT BEEN DUPLICATED.

CDRL: CONTRACTOR DATA IS REQUIRED. CONSULT WITH THE COMMUNICATIONS LONG LINES SPECIALIST FOR THE PROPER CDRL ENTRIES.

Figure VIII. Sample Output from the PC Based Expert System

3.1.13 COMMUNICATIONS LONG LINES

** DOCUMENT CALL **

DCAC Circular 310-130-1	Submission of Telecommunications
Apr 76	Service Requests
	ALL

** ESSENTIAL ENTRY **
Requests for leased long lines to support this system must be submitted to the procuring activity not later than 150 days prior to the need date.

The following output is found on a different listing.

** ACTION MESSAGE **
SOW: COORDINATE WITH SYSTEMS ENGINEERING TO ENSURE THAT THIS REQUIREMENT HAS NOT BEEN DUPLICATED.

CDRL. CONTRACTOR DATA IS REQUIRED. CONSULT WITH THE COMMUNICATIONS LONG LINES SPECIALIST FOR THE PROPER CDRL ENTRIES.

Figure IX. Sample Output From the Mainframe Version of CGADS

format. The CGADS system is formatted by task. Therefore, the paragraph numbers will differ as shown in Figures VIII and IX.

Another difference between the two systems involves theway the questions are asked. The mainframe version asks all YES-NO questions. The PC version besides asking YES-NO questions also asks multiple choice questions which were developed by combining several of the YES-NO questions in the mainframe version into one question with several possible responses. Also, the PC version will answer several questions for you depending on how you answered the previous questions. Overall, the PC version asks fewer questions and produces relatively the same data as the mainframe version. Also, the PC version has a section which asks questions and produces output for Test and Evaluation which is not present in the mainframe version. Finally, the PC version has the output organized in a Work Breakdown Structure as called out in MIL-STD-245B which differs from the organization of the output produced by the mainframe version of CGADS.

Another difference between the PC version and mainframe version is the text the systems use to build the output files. The mainframe version has the text either compiled within its code or in a database system. The PC version has the text in an ASCII file independent of the expert system code. The text for the PC system can be easily updated to the way a particular SPO does business through the use of a word processor. The mainframe version can not be easily updated by a SPO for

the way they do business and must rely on the organization in charge of the system to update the text which may not be as frequent as the SPO likes.

Conclusion

This research project can be considered a success. The resulting expert system was considered a useful product by both experts and nonexperts in SOW development.

An interview with an expert in SOW development along with the use of a model for determining whether an expert system is appropriate for a particular application determined that an expert system can be used for this project. The assistance of Rome Air Development Center in developing the rules and text made this project successful. Also, the availability of KnowledgePro with its capabilities made the development effort easier. A program which helps in the development of a SOW and is more readily accessible on a PC computer as opposed to being on a mainframe computer resulted. Finally, the methodology used in this project can be used to develop an expert system to assist in developing other contractual documents.

V. Conclusion and Recommendations

Overview

This chapter summarizes the research's objective, methodology, findings, and recommendations for further research.

Objective

The objective of this research was to build a PC version of CGADS using an expert system shell. More specifically, the portion of CGADS which assists in the development of the Full Scale Development Statement of Work.

Methodology

The methodology used to accomplish the research objective followed the steps in developing a small scale expert system discussed in the literature review and chapter III. An expert in developing a Statement of Work was interviewed using the model shown in Appendix A. The interview determined that an expert system was appropriate in assisting SOW developers write a SOW. The expert system shell chosen for the project was KnowledgePro Version 1.4. The majority of the rules were obtained from Rome Air Development Center. Any holes were filled in by studying the operation of the present CGADS system. The rules were then redesigned into Work Breakdown Structure as called out in MIL-STD-245B. The rules were also coded into the KnowledgePro programming language. The system was tested, expanded and revised by comparing it with the

operation of the present CGADS system. Finally, the system can be easily updated and maintained with the help of a wordprocessor and KnowledgePro development system. The wordprocessor is all that is needed to update the output text file, the main program text files, and the rules.

Overall the methodology used worked out well. But, problems were found with the model in Appedix A. Several of the questions were found to be redundant and unnessary. For example, question 11 is answered by the Justification section and question 16 is about the same as question 1. Also, the questions need to require more than a straight Yes or No response to make the model more interesting. For example, using some scale of measure to determine what percentage of the experts agree on a particular solution.

KnowledgePro proved to be a good software package to use for this project mainly because of its hypertext capabilities. There are several software packages that are better designed for the development of an expert system but do not provide better hypertext capabilities as well as KnowledgePro.

KnowledgePro is not without its problems. Several expert system package provide a better capability of passing parameters between modules. These expert systems have special commands that provide this capability. In KnowledgePro, additional code must be written to write each variable to a file and then commands to read each variable from a file. If the value of the variable changes at any time, additional code must be written to delete the old data before writing the new

data to the file. Also, KnowledgePro does not lend itself to data manipulation too easily. KnowledgePro does not have the capability of manipulating the data into a different format if so desired. For example, this project writes data out to a file. The data is in a logical order only if the modules are executed in a natural sequence. If the user does something other than this sequence, the paragraphs in the output will not be in a numbered sequence by paragraph number. The programming language of KnowledgePro does not have the capability by itself of reorganizing the data back into a logical sequence without interfacing the system with a programming language.

Part of the validation effort included comparing the output of the expert system with the output of the present CGADS system for several combination of responses. Overall, the outputs appeared to be close to the same. But, due to the large number of possible combinations it was impractical to compare the systems for all possible combinations. The systems may differ for some combinations of responses. Also, the expert system is a redesign of the present CGADS system and it may not have all the same questions as CGADS. But on the other hand, the expert system covers areas the CGADS system does not like System Test and Evaluation.

Research Benefits

This research project turned out to be a success. The benefits proposed in chapter I of this thesis document were

later proven in the demonstration and validation portions of this research project. These benefits include the ability to assist managers in the development of a Statement of Work (SOW). A common problem found in interviewing several managers was that paragraphs are incorporated into the SOW without any reason. The system developed for this research project provides that capability through the questioning process. Therefore, a reduction in over specification and contract costs is possible.

This system was developed for use on a microcomputer which makes accessibility better than being on a mainframe computer. The System Program Office managers liked the capability to change the text to the way they do business. The text is kept in a ASCII file seperate from the main program source code and can be changed, updated and maintained through the use of a word processor. The text is not compiled with the source code. Overall, several managers showed interest in adapting this research project for use in their SPO.

The benefits of this research project is not limited to the development of a Statement of Work. The methodology used in this research project can be applied to other documents like the development of contracts, item specifications, Contract Data Requirement Lists (CDRLs), Program Management Plans (PMPs), Test and Evaluation Master Plans (TEMPs), and Acquisition Plans.

Further Research

This research project concentrated on the Full Scale Development Statement of Work. The present CGADS system produces a draft SOW for all phases of the development effort. Additional research can be performed to develop an expert system on a PC to produce draft SOW for other phases of the Development effort as discussed in the literature review.

Another possible research project is to adapt the system design in this research project to a way a particular SPO does business. This would require a considerable time and effort on the researchers part and the experts in the SPO.

Finally, the system developed for this research project could be enhanced. The present CGADS system allows the user to repeat a question once it is finished. The PC expert system presently does not have that capability. Another possible enhancement is to have the system tell the user which paragraph's have been accomplished. Additional questions could be asked and the text in the text files could be added, updated, and improved. The program could be enhanced to the point where the user can develop a template which adapts the system for a particular application. The program will read this template to answer the questions. This will reduce the number of questions asked to the user and possibly save time. Finally, interface the system with another programming language which would reorder the output into a numbered sequence by paragraph number if the SOW paragraphs are not developed in a numbered order per the main menu.

Appendix A: Preliminary Selection Criteria

This preliminary selection criteria provides information on the development of expert systems. This criteria was developed by Capt. Hazen in his thesis "How Can Air Force Civil Engineers Use Expert Systems". It is to be used to determine whether a particular problem area might be helped by an expert system.

In order to recommend the use of an expert system, all answers must be yes (except of the Justified section and questions 2 & 15). The justified section requires one yes response.

Realistic

1. Will the expert system tackle problems managers have tackled before?

If the problem currently cannot be solved, an expert system will not help (7:73).

2. Is the task in a volatile knowledge area?

Because of the development time, expert systems are generally developed only in mature knowledge areas (7:73).

Justified

3. Does the task have a high payoff?

The extensive development costs must have a substantial offset or payoff to be a justified venture for an organization (7:73,19:130).

4. Is the human expertise scarce or being lost?

The expert system can be justified scarce expertise or expertise being lost (7:73,19:130).

5. Is the expertise needed in many locations at once?

Expert systems can easily be reproduced to be used at many locations at once (7:73,19:130).

6. Is the expertise needed in a hostile environment?

Expert systems can be risked in environments of high risk, without risking the experts (7:73,19:130).

Expertise

7. Do genuine experts exist?

If genuine experts do not exist, development of the system may be very difficult or impossible (7:74,19:129).

8. Do the experts agree on solutions?

If the experts do not agree on the solutions, then the expert system may be of little or no use (7:74,19:130).

9. Are experts better than novices at performing the task?

If a novice can perform as well as an expert, then there is probably no advantage in developing an expert system (7:74,14:27).

10. Can the experts articulate their methods?

Ultimately, the experts' methods must be represented by the knowledge base within the expert system. If the experts cannot articulate their methods, the knowledge base cannot be constructed (7:74,19:129).

Task

11. Does the task have a practical value?

The task should cover enough information that users would be interested in using an expert system for assistance (7:74,19:130).

12. Does the task use heuristic solutions versus algorithms?

Heuristic solutions are based on strategies and rules of thumb, based on incomplete or uncertain information. Algorithmic solutions use exhaustive methods based on theoretical, mathematical, and/or emperical techniques. Expert systems work best with heuritic information (7:74, 19:130).

13. Can the task be manipulated symbolically?

This deals with the type of problem. Another way to consider the same question is the telephone test. Could an expert, given the necessary time, guide a novice through the problem? If so, the task could probably manipulated symbolically (7:74,19:130).

14. Do the problems encountered share some of the same common characteristics?

Problem that the expert system will solve should have some common characteristics or data inputs to the system. If they do not, the development of an expert system will be very difficult (7:75).

15. Are conventional programming techniques satisfactory?

Expert systems are often thought of as a last resort because of the extensive time and cost invested into the development effort. If no other programming technique will do all that is required or come close, then an expert system may be the answer (7:75,14:27).

16. Is the task well understood?

This question complements Question 10. If the task is so new or so poorly understood that it requires basic research to

find solutions, knowledge engineering will not work (7:75,19:129).

17. Can the skill required be taught to novices?

If the skill cannot be taught to novices, it is unlikely that it can be developed into an expert system (7:75,14:29).

18. Is the task of a manageable size?

The task should be sufficiently narrow and self-contained. Another way to think about it is, the expert system should be the expert for a limited task within the knowledge domain (7:75,19:132).

19. Are cases available to develop and verify the validity of the system?

The development and verification of the expert system depends on having cases illustrating the problem encountered to the knowledge engineer (7:75, 14:29).

Other Considerations

20. Is the task difficult enough to require an expert?

The problem should not be too easy, otherwise the knowledge could be transferred directly to the user (7:75, 19:132).

21. Does the task require commonsense?

Expert sytems do not deal well with commonsense reasoning problems (7:75,19:129).

22. If commonsense is required, can the user provide it?

If the user can provide the commonsense required for the problem, then the expert system would be of use (7:75,19:129).

23. **Does the task require cognitive skills only?**

If senses, such as a refined sense of smell, are required, then the user will have to supply these skills (7:75,19:129).

Appendix B: Program Documentation

Program Organization

The PC CGADS system has been written with Knowledge Garden's KnowledgePro expert system shell, using KnowledgePro programming language. The source code is included in Appendix D. A software tree showing the organization of the software is in Appendix C.

Program Design

The amount of memory available in IBM-PC or compatible computers limited the code present in computer memory at one time. In order for the program to run, the software had to be broken up into seperate and independent modules. The modules are loaded into memory when called by a menu or submenu and removed from memory when finished executing. Figure X lists each module and its purpose in the overall program. The source code is kept in files with the .kb suffix, the compiled code has a file suffix of .ckb, and the text files have a .txt suffix. For every .kb file there is a .ckb file and .txt file with the same prefix (e.g. design.kb, design.ckb, and design.txt).

In order to incorporate inferencing in the the system, data must be passed between modules. Parameter passing is performed by having the originating module write to a file with the same prefix but with a .dat suffix (e.g. design.ckb writes to design.dat). The modules which needs that data then

Software module	Text Produced
Commun.kb	3.5.1.6 Communications
Data.kb	3.6 Data
Design.kb	3.5.1.1 Design Engineering
Facility.kb	3.7 Facilities
Logistic.kb	3.5.1.2 Logistics Engineering
Menu.kb	Produces MAIN MENU
Menu_Sys.kb	System/Project Management SUBMENU
Security.kb	3.5.1.5 Security
Spares.kb	3.10 Initial Spares and Repair Parts
Special.kb	3.5.1.3 Specialty Engineering
Supequip.kb	3.8 Support Equipment
Support.kb	3.4.1 Test and Evaluation Support
Testeval.kb	3.4 Test and Evaluation
Training.kb	3.3 Training

Figure X. Software Module Description

reads necessary data from the .dat file. A .dat file may look as follows:

```
Design_Change
A
/
```

Design_Change is the name of the variable being passed. "A" is the response to a question the variable represents. and "/" is an end marker used by the command READ to determine the end of a block of text. Therefore, by using the command "Design_Change is read('design.dat', 'Design_Change','/')" the variable Design_Change in the calling module will have the value of A. Figure XI contains a list of modules which use .dat files, which .dat files they use and which variable are obtained from the .dat files. Also, note that if a module looks for a .dat file and it does not exist the variable is initialized to UNKNOWN and the module executes without any problems.

A block of text is read using the same concept as reading data from a .dat file except the command FIND is used. Each module except the modules containing the menus have a topic called Include_Paragraph. A topic can best be compared to a subroutine in a common programming language. Include_Paragraph is passed the subject area and a string of text to find in an ASCII file. The command FIND is used to locate that block of text and returns the string and all the text underneath it except the end of block marker. This text is then displayed

Receiving module	variable passed	Source module
Supequip.kb	Type_Equipment	Design.kb
Data.kb	Design_Change	Design.kb
	Warranty_Maintenance	Logistic.kb
	Type_Prod_Contract	Pmgmt.kb
Facility.kb	GFE_Maintenance	Support.kb
Testeval.kb	Design_Change	Design.kb
Special.kb	Prime_Mission_Equipment	Design.kb
	NDI_ITEMS	Logistic.kb
Training.kb	Warranty_Maintenance	Logistic.kb
	Type_Equipment	Design.kb
Pmgmt.kb	Software_Test_Program	Testeval.kb
	Design_Change	Design.kb
	Warranty_Maintenance	Logistic.kb
	NDI_ITEMS	Logistic.kb
Special.kb	Prime_Mission_Equipment	Design.kb
	NDI_ITEMS	Logistic.kb

Figure XI. Variables Passed Between Software Modules

in a window and written out to a file. The topic area which was passed to the topic Include_Paragraph is displayed as the window title for the block of text which was just obtained. An example of a .txt file is found in Figure XII. The symbols '//' mark the beginning of a block and '/end' marks the end of the block. The text next to the '//' is the string of text the FIND command locates. This string of text consists of a variable and the information this variable represents (e.g. Design_Change-YES, Design_Change is the variable and YES is the user's response to the question the variable represents).

Modules which write data to a .dat file also have a topic called 'Topic Delete_Data'. This topic deletes the old data in the .dat file by deleting the entire file, recreates the .dat file and initializes the .dat file by closing it. If the file was not closed, the WRITE command may not be able to write out to the file.

Installation

The expert system requires an IBM-PC or compatible computer and a minimum of 1.75MB of hard disk space. The runtime version of KnowledgePro requires 500K bytes of disk space. The source code (.kb files) requires 118K bytes of disk space. The compiled code (.ckb files) requires 343K bytes of disk space. The text used by the expert system (.txt files) requires 366K bytes of disk space. Also, allow 250K bytes of disk space for the output document. If disk space is limited,

```
//NDI_Items-UNKNOWN
** ACTION MESSAGE **
OTS/COTS is an FSD or production type effort and should be
considered under this phase.
/end

//GFE_Required-Some_GFE
** ESSENTIAL ENTRY **
SOW:  GFE/GFP system safety analysis, MIL-STD-882B, task 213,
...(tailored task statement).

** ACTION MESSAGE **
SOW:  TASK 213 OF MIL-STD-882B MUST BE TAILORED BY YOU AND THE
SSM THEN INCORPORATED INTO THE SOW.
/end

//GFE_Required-NO_GFE
** ACTION MESSAGE **
THIS SYSTEM WILL NOT CONTAIN ANY GFE/GFP THAT WILL INTERFACE
DIRECTLY WITH THE CONTRACTOR DEVELOPED HARDWARE OR SOFTWARE.
/end

//GFE_Required-UNKNOWN
Note:  MIL-STD-882B, Task 213, GFE/GFP system safety analysis,
will require the contractor to identify the safety critical
perforance and design data needed to incorporate the GFE/GFP
items.

** ACTION MESSAGE **
CONTACT YOUR PROGRAM PERSONNEL FOR DETERMINATION.  CHECK WITH
THE SYSTEM SAFETY OFFICE FOR GUIDANCE.
/end

//Safety_Test-YES
** ESSENTIAL ENTRY **
SOW:  Safety verification, MIL-STD-882B task 207, ...(tailored
task statement)

CDRL:  DI-SAFT-80102, safety assessment report - safety
verification.
** ACTION MESSAGE **
SOW:  TASK 207 OF MIL-STD-882B MUST BE TAILORED BY YOU AND THE
SSM THEN INCORPORATED INTO THE SOW.

CDRL: TAILOR DID IAW THE PREFERRED DATA LIST.
/end
```

Figure XII. Example of a .txt File (From Specialty.txt)

the output can be sent to a floppy disk and the .kb files are not required for the operation of the expert system. If the KnowledgePro development system is being used instead of the runtime version, allow another 1MB of disk space.

To install the system onto hard disk insert the knowledgePro disk into drive A and type

a:install Enter

Follow the directions displayed on the screen to continue the installation of the system. When the request for a Programs disk is displayed, hit control C to terminate the execution of the batch file. Answer Yes to the following question:

Terminate Batch Job? [Y or N]

The installation batch file creates a directory called GARDEN and unarchives the archived files into the GARDEN directory. When the batch job is complete, the system is left in the GARDEN directory. Then copy into the GARDEN directory all the files on the diskette containing the object code (the files with the .ckb suffix) including the file kp.bat. Also, copy all the files on the text diskette (.txt files). The copying in of the source code (.kb files) is optional.

Prior to installation, make sure FIND.ARC is on the runtime version diskette and the command "ARCE FIND.ARC" is included in the installation batch file. Also, the kp.bat file must be edited to duplicate the kp.bat source code in Appendix D. Successful execution of this system requires the use of the FIND command which must be installed into the system.

Program Execution

Once the software is installed, type "kp". The screen should look as follows:

C:\GARDEN>kp

The command 'kp' starts the KnowledgePro batch file starts the execution of KnowledgePro. Once KnowledgePro is executing, the KnowledgePro main menu listing all the expert system files is displayed. Select the file title "menu" from the main menu by moving the highlight bar via the arrow keys, a mouse, or typing M on the keyboard. Once "menu" is highlighted, hit carriage return and the module "menu.ckb" will be executed. First a question is displayed on the screen asking if a new document is under development. Please note the warning message displayed on the screen along with the question. After making appropriate selection, the system asks for the name of the output file. Again note the warning displayed along with the message. Once the file name is inputted, hit a carrage return and the system will display several pages of instructions. To read the instructions, use the arrow keys to move up or down the document. Once finished reading the instructions, hit the space bar to display the main menu. Select the paragraph from the main menu you wish to develop. Note that the selection '3.5.2 System/ Project Management' will lead you to a submenu displaying the selections '3.5.2.1 System Management', '3.5.2.2 Project Mangagement', HELP, and Quit. The '3.5.2.1 System Management' selection will lead you to another submenu. To return to a previous menu, select

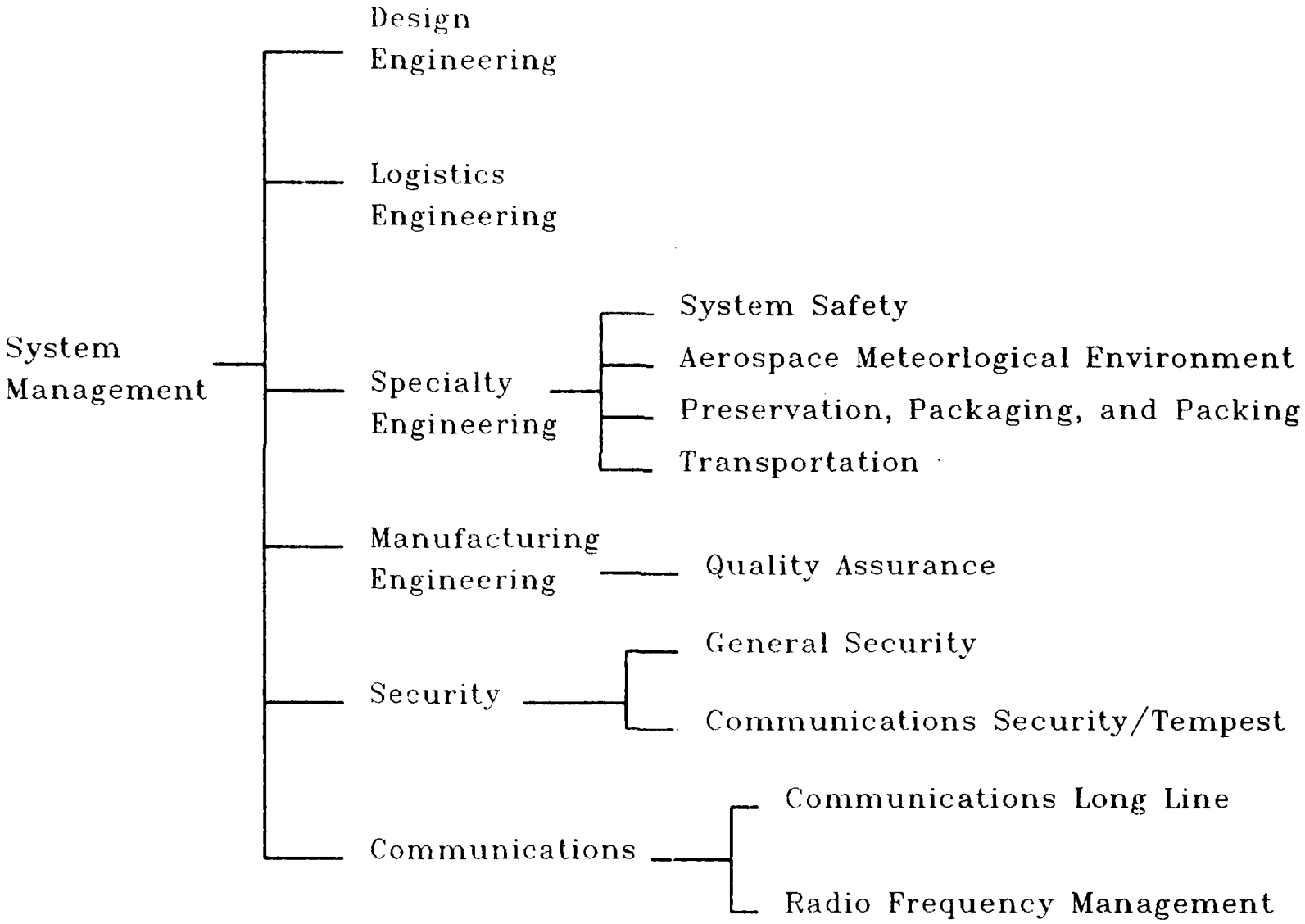
'QUIT' or 'NONE OF THE ABOVE' depending on the submenu. If you want to quit the program entirely, press F10 and you will be returned to the KnowledgePro main menu.

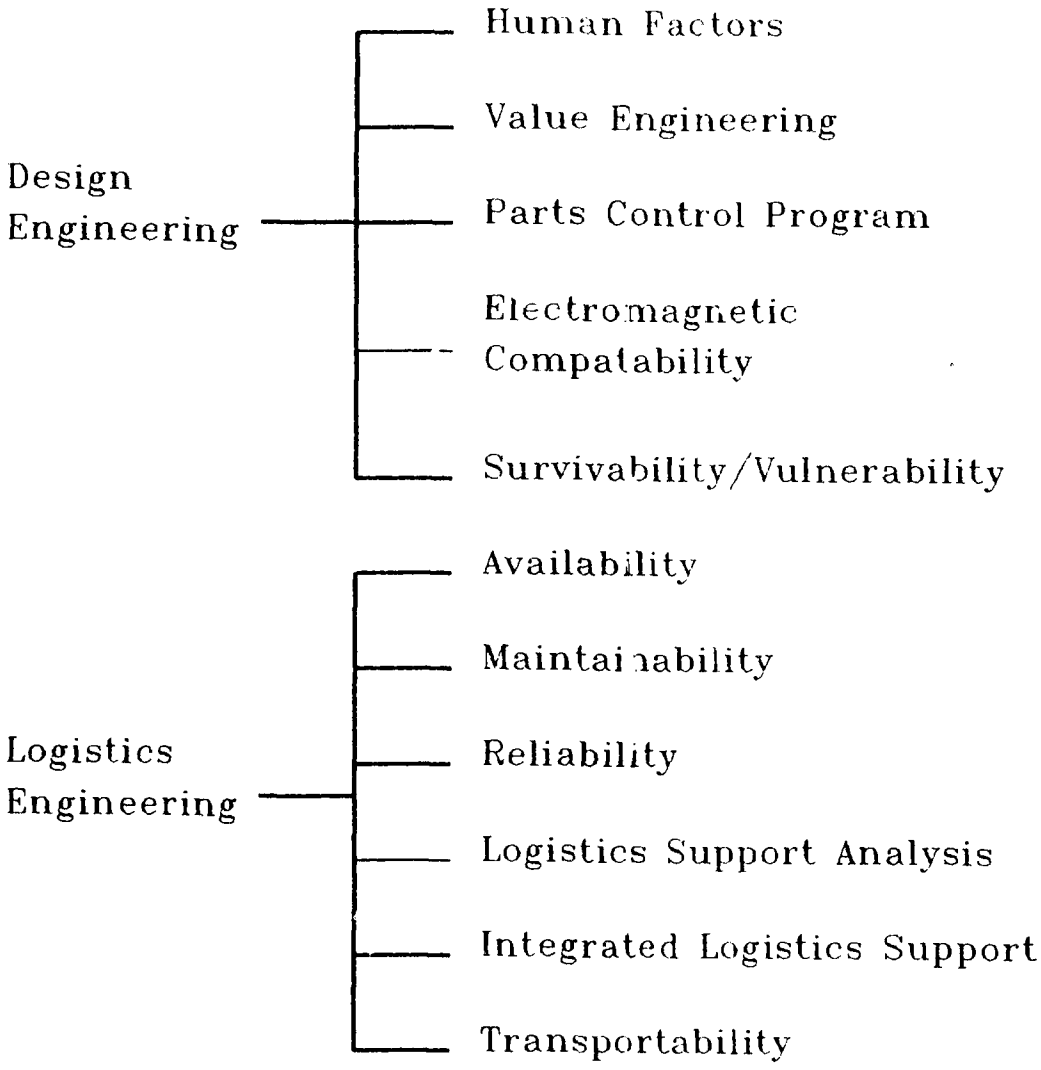
Also note that the text is written into the output file in the order you make your menu selections. If you fail to input an output file name at the beginning of the program, select the menu selection 'Change output file'. This selection can also be made to change the output file periodically throughout the execution of the program, since the output file can get large and certain word processors might not be able to load in a large file. If you completely fail to input a file name, an error message is displayed everytime the program tries to write text to file. The error message can be removed from the screen by pressing the Space bar.

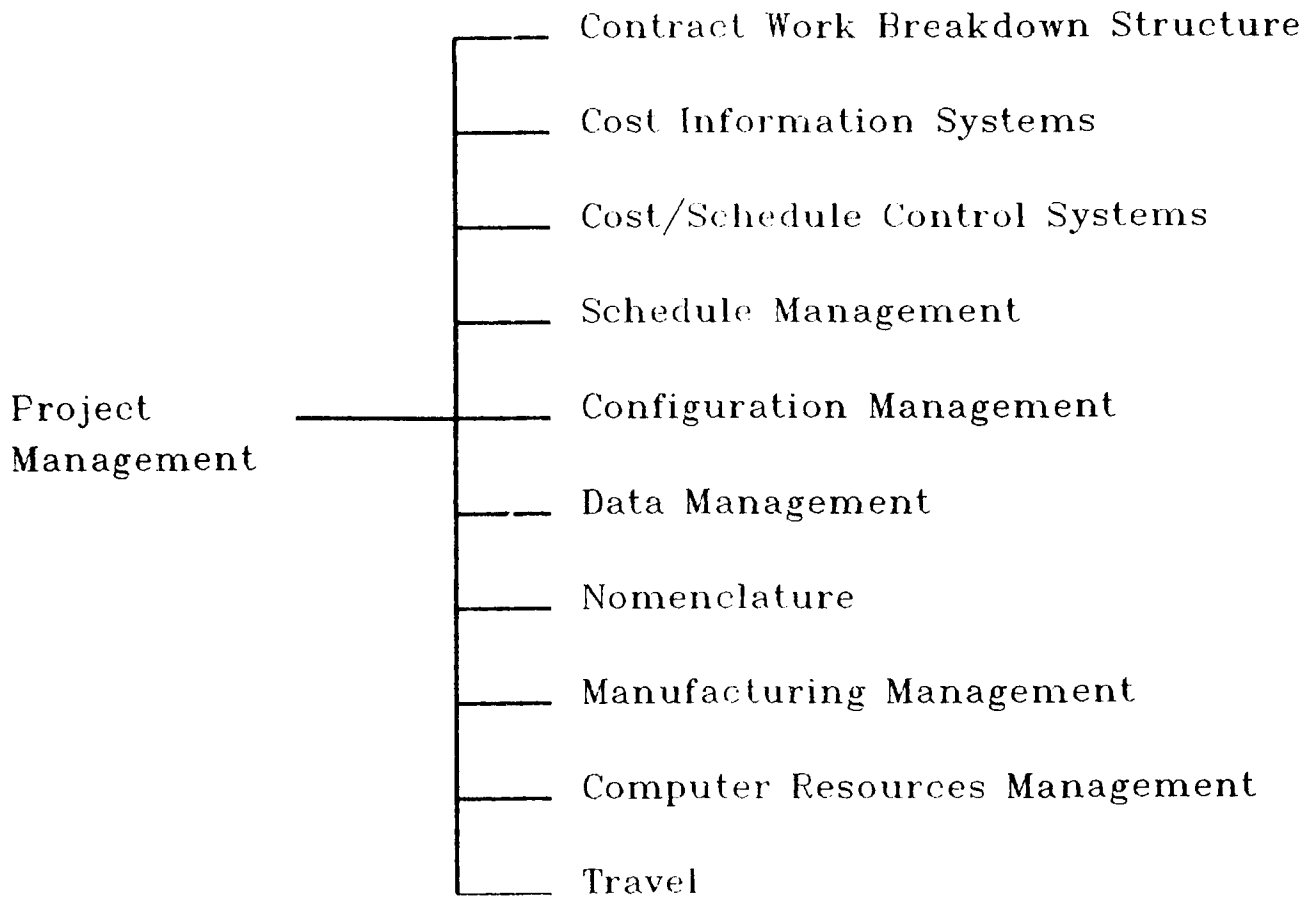
The KnowledgePro package includes a text editor. So, if at anytime you wish to edit a text file, go to the KnowledgePro main menu and press F7. The system will then ask for a file name. Input the file name and hit carriage return. When finished editing the file, hit ESC to save the file or F10 to exit the editor without saving the edited text.

To exit into DOS without leaving the KnowledgePro environment, press F8. To return to the KnowledgePro environment, type EXIT and a carriage return. To leave the KnowledgePro environment, go to the KnowledgePro main menu page and press F10.

Appendix C: Software Tree


System Management
Design Engineering
Logistics Engineering
Specialty Engineering
System Safety
Aerospace Meteorlogical Environment
Preservation, Packaging, and Packing
Transportation
Manufacturing Engineering
Quality Assurance
Security
General Security
Communications Security/Tempest
Communications
Communications Long Line
Radio Frequency Management


Design Engineering
Human Factors
Value Engineering
Parts Control Program
Electromagnetic Compatability
Survivability/Vulnerability
Logistics Engineering
Availability
Maintainability
Reliability
Logistics Support Analysis
Integrated Logistics Support
Transportability


Project Management
Contract Work Breakdown Structure
Cost Information Systems
Cost/Schedule Control Systems
Schedule Management
Configuration Management
Data Management
Nomenclature
Manufacturing Management
Computer Resources Management
Travel

Appendix D: Source Code

```
(*                    COMMUN.KB                                    *)
(* COMMUN.KB PRODUCES THE TEXT FOR SOW PARAGRAPH -                 *)
(* 3.5.1.6 COMMUNICATIONS                                          *)

ASK('INPUT THE OUTPUT FILE NAME.',?outfile).
do(Communications).

(* TOPIC COMMUNICATIONS IS THE PARENT TOPIC                       *)
Topic Communications.
write([?outfile],'3.5.1.6 Communications',#n).
do(Restore).

Topic Restore.
EOF is number_to_char(26).
data is find('Design.dat').
IF ?data <> ?EOF
THEN
Prime_Mission_Equipment is
read('Design.dat','Prime_Mission_Equipment','/')
ELSE Prime_Mission_Equipment is UNKNOWN.
do(Long_Line_Communications).
end. (* Restore *)


Topic Long_Line_Communications.
write([?outfile],'COMMUNICATIONS LONG LINE:',#n).
area is 'COMMUNICATIONS LONG LINE'.

ASK ( 'Will the system being procured require leased
long-line communications, intercommecting services,
or support?', Long_Line_Communications, [YES, NO, UNKNOWN]).

IF ?Long_Line_Communications is YES
THEN Include_Paragraph(?area,'Long_Line_Communications-YES').
do(Prime_Mission_Equip).
End. (*Long_Line_Communications*)

Topic Prime_Mission_Equip.
IF ?Prime_Mission_Equipment <> UNKNOWN
THEN Prime_Mission_Equip is Prime_Mission_Equipment AND
     do(Electromag_Radiation)
ELSE
ASK('This contract requires:

A.   PRIME MISSION EQUIPMENT FOR A RADAR, COMMUNICATIONS, OR
     NAVIGATIONAL SYSTEM THAT USES THE ELECTROMAGNETIC
     SPECTRUM.
B.   PRIME MISSION EQUIPMENT NOT FOR A RADAR.
```

```
    COMMUNICATIONS, OR
    NAVIGATIONAL SYSTEM THAT USES THE ELECTROMAGNETIC
    SPECTRUM.
C.  NO PRIME MISSION EQUIPMENT
D.  UNKNOWN.',
Prime_Mission_Equip,[A,B,C,D]).
do(Electromag_Radiation).
end. (* Prime_Mission_Equip *)

Topic Electromag_Radiation.
write([?outfile],'RADIO FREQUENCY MANAGEMENT:',#n).
area is 'RADIO FREQUENCY MANAGEMENT'.

IF ?Prime_Mission_Equip is A
THEN Electromag_Radiation is YES
ELSE
ASK ('Does this contract involve any device that radiates
or receives electromagnetic energy, including off-the-shelf
equipment?', Electromag_Radiation, [YES, NO, UNKNOWN]).

IF ?Electromag_Radiation is YES
THEN Include_Paragraph(?area,'Electromag_Radiation-YES').

IF ?Electromag_Radiation is UNKNOWN
THEN Include_Paragraph(?area,'Electromag_Radiation-UNKNOWN').
End. (* Electromag_Radiation *)

Topic Include_Paragraph(area,findstring).
Text is find('commun.txt',,,?findstring,'/end').
window(?area,,,,1,7,78,13).
say(?text).
write([?outfile],?text,#n).
close_window().
close('commun.txt').
end. (* Include_Paragraph *)
END. (* Communications *)
```

```
(*                    DATA.KB                          *)
(* DATA.KB PRODUCES THE TEXT FOR PARAGRAPH - *)
(* 3.6 DATA                                           *)

ASK('input file name',outfile).
do(Data).

Topic Data.
Write([?outfile], '3.6 DATA',#n).
do(Restore).

Topic Restore.
EOF is number_to_char(26).
data is find('Design.dat').
IF ?data <> ?EOF
THEN Design_Change is read('Design.dat','Design_Change','/')
ELSE Design_Change is UNKNOWN.

data is find('logistic.dat').
IF ?data <> ?EOF
THEN Warranty_Maintenance is read('logistic.dat',
'Warranty_Maintenance','/')
ELSE Warranty_Maintenance is UNKNOWN.

data is find('Pmgmt.dat').
IF ?data <> ?EOF
THEN Type_Prod_Contract is read('Pmgmt.dat',
'Type_Prod_Contract','/')
ELSE Type_Prod_Contract is UNKNOWN.
do(Technical_Orders).
end. (* Restore *)

Topic Include_Paragraph(area,findstring).
Text is find('data.txt',,,?findstring,'/end').
window(?area,,,,1,7,78,13).
say(?text).
write([?outfile],?text,#n).
close_window().
close('data.txt').
end. (* Include_Paragraph *)

        (*   TECHNICAL ORDERS *)

Topic Technical_Orders.
write([?outfile],'TECHNICAL ORDERS:',#n).
area is 'TECHNICAL ORDERS'.
ASK('Technical Orders will:', Technical_Orders, ['Follow
Mil-Spec',
    'Not Follow Mil-Spec', UNKNOWN]).

IF ?Technical_Orders is 'Follow Mil-Spec'
```

```
THEN Include_Paragraph(?area,'Technical_Orders-Follow_Mil_Spec')
and
     do(To_Deliverables).

IF ?Technical_Orders is UNKNOWN
THEN do(To_Deliverables).
end. (* Technical_Orders *)

Topic To_Deliverable
ASK('This program will require:
A.  A TECHNICAL ORDER PUBLICATION PLAN
B.  TECHNICAL ORDER STATUS AND SCHEDULES
C.  TECHNICAL ORDER REVIEWS
D.  ALL THE ABOVE
E.  A TECHNICAL ORDER PUBLICATION PLAN & TECHNICAL ORDER STATUS
    AND SCHEDULES
F.  A TECHNICAL ORDER PUBLICATION PLAN & TECHNICAL ORDER REVIEWS
G.  TECHNICAL ORDER STATUS AND SCHEDULES & TECHNICAL ORDER
REVIEWS.
H.  NONE OF THE ABOVE
I.  UNKNOWN',
To_Deliverables, [A,B,C,D,E,F,G,H,I]).

IF ?To_Deliverables is A
THEN Include_Paragraph(?area,'To_Deliverables-A').

IF ?To_Deliverables is B
THEN Include_Paragraph(?area,'To_Deliverables-B').

IF ?To_Deliverables is C
THEN Include_Paragraph(?area,'To_Deliverables-C').

IF ?To_Deliverables is D
THEN Include_Paragraph(?area,'To_Deliverables-A') and
     Include_Paragraph(?area,'To_Deliverables-B') and
     Include_Paragraph(?area,'To_Deliverables-C').

IF ?To_Deliverables is E
THEN Include_Paragraph(?area,'To_Deliverables-A') and
Include_Paragraph(?area,'To_Deliverables-B').

IF ?To_Deliverables is F
THEN Include_Paragraph(?area,'To_Deliverables-A') and
     Include_Paragraph(?are    o_Deliverables-C').

IF ?To_Deliverables is G
THEN Include_Paragraph(?area,'To_Deliverables-B') and
     Include_Paragraph(?area,'To_Deliverables-C').

IF ?To_Deliverables is H
THEN Include_Paragraph(?area,'To_Deliverables-H').
```

```
do(Hardware).
end. (* To_Deliverables *)

(*  ENGINEERING DATA *)

Topic Hardware.
write([?outfile],'ENGINEERING DATA',#n).
area is 'ENGINEERING DATA'.
IF ?Design_Change is Hardware OR
   ?Design_Change is Both
THEN Hardware is YES
ELSE
     IF ?Design_Change is Software
     THEN Hardware is NO
ELSE
     ASK('Are hardware items being developed and/or
          selected?', Hardware, [YES, NO, UNKNOWN]).

IF ?Hardware is YES or
   ?Hardware is UNKNOWN
THEN do(Engineering_Data_Required)
ELSE do(Data_Management).
end. (*  Hardware *)

Topic Engineering_Data_Required.
IF ?Warranty_Maintenance is C OR
   ?Type_Prod_Contract is Competitive OR
   ?Type_Prod_Contract is Sole_Source
THEN Engineering_Data_Required is YES
ELSE
ASK('Are ANY of the following statements true about this
contract?

(1) The Production Effort is to be completed
               OR
(2) The ALC will be procuring/providing spares.
               OR
(3) The system/equipment will be AF maintained/supported.
               OR
(4) There will be a follow-on Production Option.',
Engineering_Data_Required, [YES, NO, UNKNOWN]).

IF ?Engineering_Data_Required is YES
THEN Include_Paragraph(?area,'Engineering_Data_Required-YES').

IF ?Engineering_Data_Required is NO
THEN Include_Paragraph(?area,'Engineering_Data_Required-NO').
do(Data_Management).
end. (* Engineering Data Required *)

(*   DATA MANAGEMENT *)
```

```
Topic Data_Management.
write([?outfile],'MANAGEMENT DATA',#n).
area is 'MANAGEMENT DATA'.
Include_Paragraph(?area,'Data_Management').
end. (*Data_Management*)


End. (* Data *)
```

```
(*                          DESIGN.KB                              *)
(* DESIGN.KB PRODUCES THE TEXT FOR SOW PARAGRAPH -                 *)
(* 3.5.1.1 DESIGN ENGINEERING                                      *)

ASK('input a file name.',outfile).
do('Systems Engineering').

(* TOPIC SYSTEMS ENGINEERING IS THE PARENT TOPIC (MAIN TOPIC)  *)
Topic 'Systems Engineering'.
write([?outfile],'3.5.1.1 Design Engineering',#n).
do(Delete_Data).

Topic Delete_Data.
(* Delete old data *)
EOF is number_to_char(26).
data is find('Design.dat').
IF ?data <> ?EOF
THEN dos('del design.dat') and
     new_file('design.dat') and
     close('design.dat').
do(Design_Change).
end.(* Delete_Data *)

Topic Design_Change.
area is 'System Engineering'.
ASK (' This contract involves the design/redesign of',
Design_Change, [Software,Hardware,Both,'No design or redesign']).

close('Design.dat').
WRITE('Design.dat',#o,'Design_Change',?Design_Change,'/').

IF ?Design_Change is Software
or ?Design_Change is Hardware
or ?Design_Change is Both
THEN  Include_Paragraph (?area,'Design_Change-YES') and
      do(SEMP_Submitted).

IF ?Design_Change is 'No design or redesign'
THEN  Include_Paragraph (?area,'Design_Change-NO') and
      do(Human_Factors).
end. (*Design_change*)

Topic SEMP_Submitted.
ASK('Is a Systems Engineering Management Plan (SEMP) required to
be submitted in conjunction with the System Engineering
Proposal?',
SEMP_Submitted, [YES,NO, UNKNOWN]).

IF ?SEMP_Submitted is YES
THEN Include_Paragraph(?area,'SEMP_Submitted-YES').
```

```
IF ?SEMP_Submitted is UNKNOWN
THEN Include_Paragraph(?area,'SEMP_Submitted-UNKNOWN').
do(SDR).
end. (* SEMP_Submitted *)

Topic SDR.
ASK (' Was a Systems Design Review completed during the
Validation Phase?', SDR, [ YES,No,UNKNOWN]).

IF ?SDR is No
THEN  Include_Paragraph(?area,'SDR-NO').

IF ?SDR is UNKNOWN
THEN  Include_Paragraph(?area,'SDR-UNKNOWN').
do(FQR).
end. (* SDR *)

Topic FQR.
ASK ('Do you intend to conduct a Formal Qualification Review?',
FQR, [ YES,No,UNKNOWN]).
IF ?FQR is YES
THEN  Include_Paragraph(?area,'FQR-YES').
do (Config_audits).
END. (*FQR*)

Topic Config_audits.
ASK (' The following audits will be conducted during
Full Scale Development:', Config_audits, [FCA,PCA,
Both,UNKNOWN]).

IF ?Config_audits is FCA
THEN  Include_Paragraph(?area,'Config_audits-FCA').

IF ?Config_audits is PCA
THEN  Include_Paragraph(?area,'Config_audits-PCA').

IF ?Config_audits is Both
THEN   Include_Paragraph(?area,'Config_audits-FCA') and
       Include_Paragraph(?area,'Config_audits-PCA').

IF ?Config_audits is UNKNOWN
THEN  Showtext(?area,'Config_audits-UNKNOWN').
do(Human_Factors).
End. (*Config_audits*)

Topic Human_Factors.
write([?outfile],'HUMAN FACTORS:',#n).
area is 'HUMAN FACTORS'.
IF ?Design_Change is 'No design or redesign'
THEN Human_Factors is NO
ELSE
```

```
ASK (' Will this project or system be operated by,
maintained by, or controlled by people?',
Human_Factors, [ Yes,No,Unknown]).

IF ?Human_Factors is YES
THEN  Include_Paragraph(?area,'Human_Factors-YES').
do(Position_Description).
End. (*Human_Factors*)

Topic Position_Description.
ASK (' Will your system require new equipment/operation
position descriptions or significant modification of old ones?',
Position_Description, [YES,NO,UNKNOWN]).

IF ?Position_Description is YES
THEN  Include_Paragraph(?area,'Position_Description-YES') AND
      do(TEST_EVAL).

IF ?Position_Description is NO
THEN do(DOLLARS).

IF ?Position_Description is UNKNOWN
THEN do(TEST_EVAL).
End. (*Position_Discription*)

Topic TEST_EVAL.
ASK (' The development test and evaluation of this
system/equipment will be performed by the:',
TEST_EVAL, [Contractor,'SPO and/or user',all_of_the_above]).

IF ?TEST_EVAL is Contractor OR
   ?TEST_EVAL is all_of_the_above
THEN  Include_Paragraph(?area,'Test_Eval-Contractor').
do(DOLLARS).
END  (*TEST_EVAL*)

Topic DOLLARS.
write([?outfile],'VALUE ENGINEERING:',#n).
area is 'VALUE ENGINEERING'.
ASK ('Is the dollar effort for this effort
greater than 100 thousand dollars?', DOLLARS, [YES,NO,UNKNOWN]).

IF ?DOLLARS is YES
THEN  Include_Paragraph(?area,'Dollars-YES')
ELSE
      IF ?DOLLARS is NO
      THEN Include_Paragraph(?area,'Dollars-NO').

do(Parts_Control).
END. (*DOLLARS*)
```

```
Topic Parts_Control.
write([?outfile],'PARTS CONTROL PROGRAM:',#n).
area is 'PARTS CONTROL PROGRAM'.
IF ?Design_Change is Hardware OR
   ?Design_Change is Both
THEN Include_Paragraph(?area,'Parts_Control-Hardware') and
     do(Prime_Mission_Equipment).

IF ?Design_Change is Software OR
   ?Design_Change is 'No design or redesign'
THEN Showtext(?area,'Parts_Control-Software') and
     do(Prime_Mission_Equipment).
end. (* Parts_Control *)

Topic Prime_Mission_Equipment.
write([?outfile],'ELECTROMAGNETIC COMPATABILITY:',#n).
area is 'ELECTROMAGNATIC COMPATABILITY'.
ASK ('This contract requires:

A.  Prime Mission Equipment for a radar, communications, or
    navigational system that uses the electromagnetic spectrum
B.  Prime Mission Equipment not for a radar, communications, or
    navigational system that uses the electromagnetic spectrum.
C.  No Prime Mission Equipment.
D.  UNKNOWN'
, Prime_Mission_Equipment, [A, B, C, D]).

close('Design.dat').
WRITE('Design.dat',#o,'Prime_Mission_Equipment',
?Prime_Mission_Equipment,'/').

IF ?Prime_Mission_Equipment is A
THEN  Include_Paragraph(?area,'Prime_Mission_Equipment-A') AND
      do(Type_Equipment).

IF ?Prime_Mission_Equipment is B
THEN Include_Paragraph(?area,'Prime_Mission_Equipment-B') AND
     do(Type_Equipment).

IF ?Prime_Mission_Equipment is C
THEN do(sv_required).

IF ?Prime_Mission_Equipment is D
THEN do(Type_Equipment).
end. (*Prime_Mission_Equipment*)

Topic Type_Equipment.
ASK (' The Prime Mission equipment also requires:',
Type_Equipment,[Simulators,Support_Equipment,Both,
'No Additional Equipment']).
```

```
If ?Type_Equipment is Simulators
THEN  Include_Paragraph(?area,'Type_Equipment-Simulators').

IF ?Type_Equipment is Support_Equipment
THEN
Include_Paragraph(?area,'Type_Equipment-Support_Equipment').

IF ?Type_Equipment is Both
THEN  Include_Paragraph(?area,'Type_Equipment-Simulators') and

Include_Paragraph(?area,'Type_Equipment-Support_Equipment').

close('Design.dat').
WRITE('Design.dat',#o,'Type_Equipment',?Type_Equipment,'/').

do(system_assembled).
End. (*Type_Equipment*)

Topic system_assembled.
ASK (' Is this proposed contract for subsystems/equipment
assembled into a system?', system_assembled, [Yes,No,Unknown]).

If ?system_assembled is YES
THEN  Include_Paragraph(?area,'system_assembled-YES').
do(Equipment_Location).
End. (*system_assembled*)

Topic Equipment_Location.
ASK ('The subsystems/equipment acquired under this
contract will be placed:', Equipment_Location, ['On Board
Aircraft','On A Ground Fixed Site', 'On A Ground Mobile Site',
Unknown]).

IF ?Equipment_Location is 'On Board Aircraft'
THEN
Include_Paragraph(?area,'Equipment_Location-On_Board_Aircraft').

IF ?Equipment_Location is 'On A Ground Fixed Site'
THEN Include_Paragraph(?area,'Equipment_Location-
On_A_Ground_Fixed_Site').

IF ?Equipment_Location is 'On A Ground Mobile Site'
THEN Include_Paragraph(?area,'Equipment_Location-
On_A_Ground_Mobile_Site').
do(sv_required).
End. (*Equipment_Location*)

Topic sv_required.
write([?outfile],'SURVIVABILITY/VULNERABILITY:',#n).
area is 'SURVIVABILITY/VULNERABILITY'.
ASK (' Is a Survivability/Vulnerability (S/V)
```

```
program required?', sv_required, [YES,NO,UNKNOWN]).

IF ?sv_required is YES
THEN  Include_Paragraph(?area.'sv_required-Yes') and
      do(sv_program_plan).

IF ?sv_required is NO
THEN  Include_Paragraph(?area,'sv_required-NO').

IF ?sv_required is UNKNOWN
THEN Showtext(?area,'sv_program_plan-D').
End. (*sv_required*)

Topic sv_program_plan.
ASK ('A Survivability/Vulnerability Program Plan was,
A. Written in a prior phase and does not require updating
B. Written in a prior phase and does require updating
C. Not written
D. Unknown', sv_program_plan, [A,B,C,D]).

IF ?sv_program_plan is B
THEN  Include_Paragraph(?area,'sv_program_plan-B').

IF ?sv_program_plan is C
THEN  Include_Paragraph(?area,'sv_program_plan-C').

IF ?sv_program_plan is D
THEN Showtext(?area,'sv_program_plan-D').
exit().
End. (*sv_program_plan*)

Topic Include_Paragraph(area,findstring).
Text is find('Design.txt',,,?findstring,'/end').
window(?area,,,,1,7,78,13).
say(?text).
write([?outfile],?text,#n).
close_window().
close('Design.txt').
end. (* Include_Paragraph *)

Topic Showtext(area,findstring).
Text is find('Design.txt',,,?findstring,'/end').
window(?area,,,,1,7,78,13).
say('NOTE: THE FOLLOWING MESSAGE IS NOT WRITTEN TO DISK',?text).
close_window().
close('Design.txt').
end. (* Showtext *)

End. (* Systems Engineering *)
```

```
(*                    FACILITY.KB                    *)
(* FACILITY.KB PRODUCES THE TEXT FOR PARAGRAPH - *)
(* 3.7 FACILITY.KB                                  *)

ASK('input a file name',outfile).
do('Real Property Facilities').

(* 'REAL PROPERTY FACILITIES' IS THE PARENT TOPIC  *)
Topic 'Real Property Facilities'.
do(Restore).
write([?outfile],'3.7 Operations/Site Activation',#n).

Topic Restore.
EOF is number_to_char(26).
Data is find('support.dat').
IF ?data <> ?EOF
Then GFE_Maintenance is read('support.dat','GFE_Maintenance','/')
ELSE GFE_Maintenance is UNKNOWN.
do(Facilities_Task_Required).
END. (* RESTORE *)


Topic Facilities_Task_Required.
write([?outfile],'REAL PROPERTY FACILITIES:',#n).
area is 'REAL PROPERTY FACILITIES'.
ASK(' Are ANY of the following statements true about this
contact?

      (1) A facility is being contructed.
                  OR
      (2) Real property equipment is being installed by the
          systeM contractor.
                  OR
      (3) The system contractor is being tasked to provide design
          criteria.',

Facilities_Task_Required, [YES,NO, UNKNOWN]).

IF ?Facilities_Task_Required is YES
THEN Include_Paragraph(?area,'Facilities_Task_Required-YES') and
     do(Facilities).

IF ?Facilities_Task_Required is UNKNOWN
THEN Showtext(?area,'Facilities_Task_Required-UNKNOWN') and
     do(Facilities).
end. (*Facilities_Task_Required*)

Topic Facilities.
ASK('The system contractor is to:
A. Design the Facilities
B. Build the Facilities
```

```
C. Both Design and Build the Facilities
D. No Design or Building is required
E. UNKNOWN',
Facilities, [A,B,C,D,E]).

IF ?Facilities is A
THEN Include_Paragraph(?area,'Facilities-A').

IF ?Facilities is B
THEN Include_Paragraph(?area,'Facilities-B').

IF ?Facilities is C
THEN Include_Paragraph(?area,'Facilities-A') and
     Include_Paragraph(?area,'Facilities-B').
do(Towers_Required).
end. (*Facilities *)

Topic Towers_Required.
ASK('Are towers (structural support towers) required to support
antenna as part of your system?', Towers_Required, [YES, NO,
UNKNOWN]).

IF ?Towers_Required is YES
THEN Include_Paragraph(?area,'Towers_Required-YES').
do(Environmental_Deliverables).
end.(*Towers_Required*)

Topic Environmental_Deliverables.
ASK('Is the system contractor tasked to produce deliverables as
part of the environmental process?', Environmental_Deliverables,
[YES, NO, UNKNOWN]).

IF ?Environmental_Deliverables is YES
THEN Include_Paragraph(?area,'Environmental_Deliverables-YES').

IF ?Environmental_Deliverables is NO
THEN Include_Paragraph(?area,'Environmental_Deliverables-NO').

IF ?Environmental_Deliverables is UNKNOWN
THEN Showtext(?area,'Environmental_Deliverables-UNKNOWN').

do(Site_Selected).
end. (*Environmental_Deliverables*)

Topic Site_Selected.
ASK('Has the site for facilities been selected?', Site_Selected.
[YES, NO, UNKNOWN]).

IF ?Site_Selected is NO
THEN Include_Paragraph(?area,'Site_Selected-NO') and
     do(GFE_OR_Utilities)
```

```
ELSE do(Access_Roads).
end. (*Site_Selected*)

Topic Access_Roads.
IF ?Facilities is A OR
   ?Facilities is D
THEN Access_Roads is NO
ELSE
ASK('Will the contractor provide access roads to the site?',
Access_Roads, [YES, NO, UNKNOWN]).

IF ?Access_Roads is YES
THEN Include_Paragraph(?area,'Access_Roads-YES').

do(GFE_OR_Utilities).
end. (*Access_Roads*)

Topic GFE_OR_Utilities.
IF ?GFE_Maintenance is A
THEN GFE_OR_Utilities is YES
ELSE
ASK('Will the Government furnish equipment or utility services
(e.g.; gas, water, electricity)?', GFE_OR_Utilities, [YES, NO,
UNKNOWN]).

IF ?GFE_OR_Utilities is YES
THEN Include_Paragraph(?area,'GFE_OR_Utilities-YES').

IF ?GFE_OR_Utilities is NO
THEN Include_Paragraph(?area,'GFE_OR_Utilities-NO').

do(TO_Government_Ownership).
end. (*GFE_OR_Utilities*)

Topic TO_Government_Ownership.
IF ?Facilities is A OR
   ?Facilities is D
THEN TO_Government_Ownership is NO
ELSE
ASK('Is the facility to be transferred from contractor ownership
to Government ownership?', TO_Government_Ownership, [YES, NO,
UNKNOWN]).

IF ?TO_Government_Ownership is YES
THEN Include_Paragraph(?area,'TO_Government_Ownership-YES').
do(O&M_Required).
end.(*TO_Government_Ownership*)

Topic O&M_Required.
IF ?TO_Government_Ownership is NO
THEN O&M_Required is NO
```

```
ELSE
ASK('Is there to be a period of time that the contractor is to
operate and maintain the real property facilities after
Government acceptance of the facilities and before the government
accepts the system?',
O&M_Required, [YES, NO, UNKNOWN]).

IF ?O&M_Required is YES
THEN Include_Paragraph(?area,'O&M_Required-YES').
do(How_Big).
end. (*O&M_Required*)

Topic How_Big.
ASK('The facility is:
A. LARGER THAN 20,000 SQUARE FEET OR ESTIMATED TO COST MORE THAN
1,000,000 DOLLARS.
B. LARGER THAN 80,000 SQUARE FEET OR ESTIMATED TO COST MORE THAN
4,000,000 DOLLARS.
C. NONE OF THE ABOVE.
D. UNKNOWN',
How_Big, [A,B,C,D]).

IF ?How_Big is A
THEN Include_Paragraph(?area,'How_Big-A').

IF ?How_Big is B
THEN Include_Paragraph(?area,'How_Big-A') and
     Include_Paragraph(?area,'How_Big-B').
end. (*How_Big*)

Topic Include_Paragraph(area,findstring).
Text is find('facility.txt',,,?findstring,'/end').
window(?area,,,,1,7,78,13).
say(?text).
write([?outfile],?text,#n).
close_window().
close('facility.txt').
end. (* Include_Paragraph *)

Topic Showtext(area,findstring).
Text is find('facility.txt',,,?findstring,'/end').
window(?area,,,,1,7,78,13).
say(?text).
close_window().
close('facility.txt').
end. (* Showtext *)
end. (* Real Property Facilities *)
```

```
(*                    LOGISTIC.KB                          *)
(* LOGISTIC.KB PRODUCES THE TEXT FOR SOW PARAGRAPH -  *)
(* 3.5.1.2 LOGISTICS ENGINEERING.                     *)

ASK('INPUT AN OUTPUT FILE NAME.',?outfile).
do('Logistics Engineering').

(* TOPIC LOGISTICS ENGINEERING IS THE PARENT TOPIC    *)
Topic 'Logistics Engineering'.
write([?outfile],'3.5.1.2 Logistics Engineering',#n).
do(Delete_Data).

Topic Delete_Data.
(* Delete old data *)
EOF is number_to_char(26).
data is find('logistic.dat').
IF ?data <> ?EOF
THEN dos('del logistic.dat') and
     new_file('logistic.dat') and
     close('logistic.dat').
do(Availability).
end. (* Delete_Data *)

Topic Availability.
write([?outfile],'AVAILABILITY:',#n).
area is 'AVAILABILITY'.
Include_Paragraph(?area,'-Logistics').
do(Maintainability_Required).
end. (* Availability *)

Topic Maintainability_Required.
write([?outfile],'MAINTAINABILITY:',#n).
area is 'MAINTAINABILITY'.

ASK('Is a Maintainability Program required?',
Maintainability_Required,
[YES,NO, UNKNOWN]).

IF ?Maintainability_Required is YES
THEN Include_Paragraph(?area,'Maintainability_Required-YES')
and do(TYPE_FSD_EFFORT).

IF ?Maintainability_Required is NO
THEN do(Reliability_Required).

IF ?Maintainability_Required is UNKNOWN
THEN do(TYPE_FSD_EFFORT).
end. (*Maintainability_Required*)

Topic TYPE_FSD_EFFORT.
ASK('What is the type of Full Scale Development (FSD) effort
```

```
required?',
TYPE_FSD_EFFORT,[Competitive, 'Single Source', 'Firm Fixed
Price', UNKNOWN]).

IF ?TYPE_FSD_EFFORT is Competitive
THEN Include_Paragraph(?area,'TYPE_FSD_EFFORT-Competitive').

do(NDI_ITEMS).
end. (*TYPE_FSD_EFFORT*)

Topic NDI_ITEMS.
ASK('Will the system have:
A. SOME NON DEVELOPMENTAL ITEMS
B. ALL NON DEVELOPMENTAL ITEMS
C. NO NON DEVELOPMENTAL ITEMS
D. UNKNOWN',
NDI_ITEMS, [A,B,C,D]).

IF ?NDI_ITEMS is A OR
   ?NDI_ITEMS is B
THEN Include_Paragraph(?area,'NDI_ITEMS-A_OR_B').

close('logistic.dat').
WRITE('logistic.dat',#o,'NDI_ITEMS',?NDI_ITEMS,'/').
do(Warranty_Maintenance).
end.(* NDI_ITEMS *)

Topic Warranty_Maintenance.
ASK('How will the system be maintained/repaired?
A. CONTRACTOR WILL MAINTAIN AND REPAIR ALL ITEMS.
B. CONTRACTOR WILL PERFORM REPAIR ONLY SOME OF THE ITEMS
C. REPAIR AND MAINTENANCE IS NOT DONE BY THE CONTRACTOR
D. UNKNOWN',
Warranty_Maintenance, [A,B,C,D]).

IF ?Warranty_Maintenance is A
THEN Include_Paragraph(?area,'Warranty_Maintenance-A').

IF ?Warranty_Maintenance is B
THEN Include_Paragraph(?area,'Wa. anty_Maintenance-B').

IF ?Warranty_Maintenance is C
THEN Include_Paragraph(?area,'Warranty_Maintenance-C').

close('logistic.dat').
WRITE('logistic.dat',#o,'Warranty_Maintenance',
?Warranty_Maintenance,'/').
do(Reliability_Required).
end.(* Warranty_Maintenance *)

Topic Reliability_Required.
```

```
write([?outfile],'REALIABILITY:',#n).
area is 'REALIABILITY'.

ASK('Is a Reliability Program required?',
Reliability_Required,
[YES, NO, UNKNOWN]).

IF ?Reliability_Required is YES
THEN Include_Paragraph(?area,'Reliability_Required-YES') and
     do(Contract).

IF ?Reliability_Required is NO
THEN Include_Paragraph(?area,'Reliability_Required-NO') and
     do(Support_Factors_Documentation).

IF ?Reliability_Required is UNKNOWN
THEN do(Contract).
end. (* Reliability_Required *)

Topic Contract.
IF ?TYPE_FSD_EFFORT is Competitive
THEN Include_Paragraph(?area,'Contract-Competitive').
do(Reliability_Parameters).
end. (* Contract *)

Topic Reliability_Parameters.
ASK('Will Contractor quantitative predictions of realiability
be
required?', Reliability_Parameters, [YES, NO, UNKNOWN]).

IF ?Reliability_Parameters is YES
THEN Include_Paragraph(?area,'Reliability_Parameters-YES').

do(Support_Factors_Documentation).
end. (* Reliability_Parameters *)

Topic Support_Factors_Documentation.
write([?outfile],'LOGISTICS SUPPORT ANALYSIS:',#n).
area is 'LOGISTICS SUPPORT ANALYSIS'.

ASK('Do you require documentation on support factors for the
new system?

NOTE:  These factors include: mobility requirements,
deployment scenerios, mission frequency, basing concepts,
service life, etc. for both peacetime and wartime.  IF this
task was performed in an earlier phase, you may wish to only
update it now.  This information is generally reported in a
use study.', Support_Factors_Documentation, [YES, NO,
UNKNOWN]).
```

```
IF ?Support_Factors_Documentation is YES
THEN
Include_Paragraph(?area,'Support_Factors_Documentation-YES')
ELSE
   IF ?Support_Factors_Documentation is NO
   THEN Include_Paragraph(?area,'Support_Factors_Documentation
           -NO')
   ELSE Include_Paragraph(?area,'Support_Factors_Documentation
           -UNKNOWN').

do(Maximize_Resources).
end. (* Support_Factors_Documentation *)

Topic Maximize_Resources.
ASK('Do you want to maximize the use of existing resources
(hardware, software, trained manpower, etc.) in the newly
designed system?', Maximize_Resources, [YES, NO, UNKNOWN]).

IF ?Maximize_Resources is YES
THEN Include_Paragraph(?area,'Maximize_Resources-YES').
do(BCS_Required).
end. (*Maximize_Resources *)

Topic BCS_Required.
ASK('Do you need a Baseline Comparison System (BCS) to
determine
support parameters or support, cost and readiness drivers for
the
new system?', BCS_Required, [YES, NO, UNKNOWN]).

IF ?BCS_Required is YES
THEN Include_Paragraph(?area,'BCS_Required-YES').
do(New_Tech_Impact).
end. (* BCS_Required *)

Topic New_Tech_Impact.
ASK('Do you want the contractor to explore new technological
advances and their impact on system support?',
New_Tech_Impact, [YES, NO, UNKNOWN]).

IF ?New_Tech_Impact is YES
THEN Include_Paragraph(?area,'New_Tech_Impact-YES').
do(Support_Design_Constraints_Doc).
end. (* New_Tech_Impact *)

Topic Support_Design_Constraints_Doc.
ASK('So you want the contractor to establish or update support
design constraints and include them in LSA documentation,
System Specifications, etc.?', Support_Design_Constraints_Doc,
[YES, NO, UNKNOWN]).
```

```
IF ?Support_Design_Constraints_Doc is YES
THEN
Include_Paragraph(?area,'Support_Design_Constraints_Doc-YES').
do(Tasks_For_Operation).
end. (* Support_Design_Constraints_Doc *)

Topic Tasks_For_Operation.
ASK('Do you want to identify the operations and support tasks
needed to keep the new equipment operating?

NOTE: These tasks should be specified for each maintenance
concept being considered, for both peacetime and wartime.  The
results of this effort should be documented in the LSA Record
(LSAR) and serve as the outline for your technical orders and
manuals.', Tasks_For_Operation, [YES, NO, UNKNOWN]).

IF ?Tasks_For_Operation is YES
THEN Include_Paragraph(?area,'Tasks_For_Operation-YES').
do(Alternative_Support_Concepts).
end. (* Tasks_For_Operation *)

Topic Alternative_Support_Concepts.
ASK('Do you want the contractor ot propose and evaluate
alternative support concepts for the system?

NOTE:  The contractor will also be required to perform and
document system tradeoffs made during the design process.
These tasks are designed to force cost, schedule, readiness
and supportability to be considered along with performance
when design decisions are made.',
Alternative_Support_Concepts, [YES, NO, UNKNOWN]).

IF ?Alternative_Support_Concepts is YES
THEN
Include_Paragraph(?area,'Alternative_Support_Concepts-YES').
do(Repair_Level_Analysis).
end. (* Alternative_Support_Concepts *)

Topic Repair_Level_Analysis.
ASK('Do you want a repair level analysis performed?',
Repair_Level_Analysis, [YES, NO, UNKNOWN]).

IF ?Repair_Level_Analysis is YES
THEN Include_Paragraph(?area,'Repair_Level_Analysis-YES').
do(Tasks_For_Source_Data).
end. (* Repair_Level_Analysis *)

Topic Tasks_For_Source_Data.
ASK('Do you want the contactor to analyze the operations and
maintenance tasks to provide source data for provisioning,
technical manuals, training, manpower lists, etc.?
```

```
NOTE:  This will provide data to identify areas which need
additional design effort to reduce support costs and improve
readiness.  It is current Air Force policy to use
MIL-STD-1388-1A, MIL-STD-1388-2A and MIL-STD-1561B as the only
authorized method of acquiring provisioning technical
documentation.', Tasks_For_Source_Data, [YES, NO, UNKNOWN]).

IF ?Tasks_For_Source_Data is YES
THEN Include_Paragraph(?area,'Tasks_For_Source_Data-YES').
do (Impact_Assessment).
end. (* Tasks_For _Source_Data *)

Topic Impact_Assessment.
IF ?Support_Factors_Documentation is YES or
   ?Maximize_Resources is YES or
   ?BCS_Required is YES or
   ?New_Tech_Impact is YES or
   ?Support_Design_Constraints_Doc is YES or
   ?Tasks_For_Operation is YES
THEN Impact_Assessment is YES
ELSE ASK('Do you want the contractor to assess the impact of
introducingthe new system?

NOTE:  This includes looking at, for example, impact on depot
workloads, provisioning, ATE availability, and manpower.',
Impact_Assessment, [YES, NO, UNKNOWN]).

IF ?Impact_Assessment is YES
THEN Include_Paragraph(?area,'Impact_Assessment-YES').
do(Impact).
end. (* Impact_Assessment *)

Topic Impact.
ASK('Have any of your previous responses made an LSA Plan
mandatory orwas an LSA Plan written during an earlier phase of
this program?',
Impact, [YES, NO, UNKNOWN]).

IF ?Impact is YES
THEN Include_Paragraph(?area,'Impact-YES').
do(Joint_Service_Program).
end. (* Impact *)

Topic Joint_Service_Program.
ASK('Is this a Joint Service program?', Joint_Service_Program,
[YES,NO, UNKNOWN]).

IF ?Joint_Service_Program is YES
THEN Include_Paragraph(?area,'Joint_Service_Program-YES').
```

```
do(ILS_Required).
end. (*Joint_Service_Program*)

Topic ILS_Required.
write([?outfile],'INTEGRATED LOGISTICS SUPPORT:',#n).
area is 'INTEGRATED LOGISTICS SUPPORT'.

ASK('Is Integrated Logistices Support (ILS) Planning
Required?'
,ILS_Required, [YES, NO, UNKNOWN]).

IF ?ILS_Required is YES
THEN Include_Paragraph(?area,'ILS_Required').
do(Design_Parameters).
end. (* ILS_Required *)

Topic Design_Parameters.
write([?outfile],'TRANSPORTABILITY:',#n).
area is 'TRANSPORTABILITY'.

ASK('Are the design parameters known?', Design_Parameters,
[YES, NO, UNKNOWN]).

IF ?Design_Parameters is YES
THEN Include_Paragraph(?area,'Design_Parameters-YES').

IF ?Design_Parameters is NO
THEN Include_Paragraph(?area,'Design_Parameters-NO').

do(Pack_Haz_Materials).
end.(* Design_Parameters *)

Topic Pack_Haz_Materials.
IF exists(Pack_Hazardous_Materials)
THEN Pack_Haz_Materials is Pack_Hazardous_Materials
ELSE Pack_Hazardous_Materials is UNKNOWN.
do(Hazardous_Materials).
end. (* Pack_Haz_Materials *)

Topic Hazardous_Materials.
IF ?Pack_Hazardous_Materials is YES
THEN Hazardous_Materials is YES
ELSE
ASK('Will the system contain any hazardous materials?',
Hazardous_Materials,
[YES, NO, UNKNOWN]).

IF ?Hazardous_Materials is YES
THEN Include_Paragraph(?area,'Hazardous_Materials-YES').

IF ?Hazardous_Materials is NO
```

```
THEN Include_Paragraph(?area,'Hazardous_Materials-NO').

do(Special_Requirements).
end. (*Hazardous_Materials *)

Topic Special_Requirements.
ASK('This contract includes items that:
A. REQUIRE SPECIAL LOADING, UNLOADING, AND MOVEMENT PROCEDURES
B. INHIBIT TRANSPORTATION
C. BOTH REQUIRE SPECIAL LOADING AND INHIBIT TRANSPORATION
D. NO SPECIAL REQUIREMENTS IDENTIFIED.
E. UNKNOWN',
Special_Requirements, [A,B,C,D,E]).

IF ?Special_Requirements is A
THEN Include_Paragraph(?area,'Special_Requirements-A').

IF ?Special_Requirements is B
THEN Include_Paragraph(?area,'Special_Requirements-B').

IF ?Special_Requirements is C
THEN Include_Paragraph(?area,'Special_Requirements-A') and
     Include_Paragraph(?area,'Special_Requirements-B').
 end. (* Special_Requirements *)

Topic Include_Paragraph(area,findstring).
Text is find('Logistic.txt',,,?findstring,'end').
window(?area,,,,1,7 78,13).
say(?text).
write([?outfile],?text,#n).
close_window().
close('Logistic.txt').
end. (* Include_Paragraph *)

end. (* Logistics Engineering *)
```

```
(*                          MENU.KB                                  *)
(* MENU.KB PRODUCES THE MAIN MENU                                    *)

do(Delete_data).
Topic Delete_data.

ASK('Are you starting a new document?

NOTE: If you wish to develop a new document from scratch,
answer YES to the above question.  If you are returning to
develop a document started at a previous session answer NO to
the above question.

***** WARNING *****
PARAMETER PASSING IS PERFORMED THROUGH THE USE OF .DAT FILES.
IF YOU ANSWER THE ABOVE QUESTION YES, ALL .DAT FILES IN THE
DIRECTORY WILL BE DELETED.  IF YOU STOP WORKING ANY TIME
DURING THIS SESSION AND INTEND TO CONTINUE WORK LATER, MAKE
SURE YOU SAVE YOUR .DAT FILES.  IF THERE ARE NO .DAT FILES IN
THE DIRECTORY, A MESSAGE "FILE NOT FOUND" WILL APPEAR
BRIEFLY ON THE SCREEN.',Delete_Data, [YES,NO]).

IF ?Delete_Data is YES
THEN dos('del *.dat').
end.

Text is find('Intro.txt',,,Introduction,'/end').
say('text).
do(outfile).
Topic outfile.
ASK('Input the name of the output file.

NOTE:  THE FOLLOWING TEXT FILES CONTAIN THE PROGRAM TEXT: DO
NOT USE THEM AS YOUR OUTPUT FILE NAME: SECURITY.TXT, DATA.TXT,
DESIGN.TXT, PMGMT.TXT, SPARES.TXT, SUPEQUIP.TXT, FACILITY.TXT,
TESTEVAL.TXT, SPECIAL.TXT, TRAINING.TXT, SUPPORT.TXT,
MANUFACT.TXT, LOGISTIC.TXT, COMMUN.TXT, DESIGN.DAT,
LOGISTIC.DAT, TESTEVAL.DAT, PMGMT.DAT, AND SUPPORT.DAT. ALSO
DO NOT USE THE EXTENTIONS .KB AND .CKB.',outfile).
do(Main_Menu).
end.

Topic Main_Menu.

ASK('
                ****************************************
                *                                      *
                *            MAIN MENU                 *
                *                                      *
                ****************************************
```

```
Select the Statement of Work Topic Area you wish to generate
from the menu. Push F10 to QUIT.', Main_Menu, ['1.0 Scope',
'3.0 Requirements','3.2  Training', '3.4  System and Test
Evaluation','3.5  System/Project Management', '3.6  Data','3.7
Operations/Site Activation','3.8  Common Support Equipment',
'3.10 Initial Spares and Repair Parts', 'Change output
file']).

IF ?Main_Menu is '1.0 Scope'
THEN text is find('intro.txt',,,INDEX,'/end') and
     write(?outfile,?text) and
     text is find('intro.txt',,,?Main_Menu,'/end') and
     say(?text) and
     write(?outfile,?text) and
     do(Main_Menu).

IF ?Main_Menu is '3.0 Requirements'
THEN text is find('intro.txt',,,?Main_Menu,'/end') and
     say(?text) and
     write(?outfile,?text) and
     do(Main_Menu).

IF ?Main_Menu is '3.2  Training'
THEN Load('Training.ckb') and
     do(Training) and
     remove_topic(Training) and
     do(Main_Menu).

IF ?Main_Menu is '3.4  System and Test Evaluation'
THEN Load('Testeval.ckb') and
     do('Test and Evaluation') and
     load('support.ckb') and
     do(support) and
     remove_topic('support') and
     do(Main_Menu).

IF ?Main_Menu is '3.5  System/Project Management'
THEN Load('Menu_Sys.ckb') and
     do('System/Project Management Menu') and
     close('Menu_Sys.ckb') and
     do(Main_Menu).

IF ?Main_Menu is '3.6  Data'
THEN load('Data.ckb') and
     do(Data) and
     remove_topic('Data') and
     do(Main_Menu).

IF ?Main_Menu is '3.7  Operations/Site Activation'
THEN load('Facility.ckb') and
```

```
     do('Real Property Facilities') and
     remove_Topic('Real Property Facilities') and
     do(Main_Menu).

IF ?Main_Menu is '3.8  Common Support Equipment'
THEN load('supequip.ckb') and
     Do('Support Equipment') and
     remove_topic('Support Equipment') and
     do(Main_Menu).

IF ?Main_Menu is '3.10 Initial Spares and Repair Parts'
THEN load('Spares.ckb') and
     do('Initial Spare/Repair Parts') and
     remove_topic('Initial Spare/Repair Parts') and
     do(Main_Menu).

IF ?Main_Menu is 'Change output file'
THEN do(outfile).

IF ?Main_Menu is QUIT
THEN Exit().

END. (* Main_Menu *)
```

```
(*                    MENU_SYS.KB                          *)
(* MENU_SYS.KB PRODUCES THE SUBMENUS WHEN                  *)
(* 3.5 SYSTEM/PROJECT MANAGEMENT IS SELECTED               *)
(* FROM THE MAIN MENU                                      *)

Topic 'System/Project Management Menu'.
ASK('System/Project Management is broken down into the
subparagraphs
3.5.1 System Management
3.5.2 Project Management.
Please choose the subparagraph you wish to work on',
management, ['3.5.1 System Management', '3.5.2 Project
Management', QUIT, HELP]).

IF ?management is '3.5.1 System Management'
THEN do('System Management Menu') and
     do('System/Project Management Menu').

IF ?management is '3.5.2 Project Management'
THEN load('Pmgmt.ckb') and
     do('Program Management') and
     remove_topic('Program Management') and
     do('System/Project Management Menu').

IF ?management is HELP
THEN do(Help_Management) and
     remove_topic('Help_Management') and
     do('System/Project Management Menu').

IF ?management is QUIT
THEN exit().

Topic 'System Management Menu'.
ASK('Choose the subparagraph under the paragraph System
Management you wish to work on.', System, ['3.5.1.1 Design
Engineering', '3.5.1.2 Logistics Engineering', '3.5.1.3
Specialty Engineering', '3.5.1.4 Manufacturing Engineering',
'3.5.1.5 Security','3.5.1.6 Communications', 'NONE OF THE
ABOVE', HELP]).

IF ?System is '3.5.1.1 Design Engineering'
THEN load('DESIGN.CKB') AND
     do('Systems Engineering') and
     remove_topic('Systems Engineering') and
     do('System Management Menu').

IF ?System is '3.5.1.2 Logistics Engineering'
THEN load('logistic.ckb') and
     do('Logistics Engineering') and
     remove_topic('Logistics Engineering') and
     do('System Management Menu').
```

```
IF ?System is '3.5.1.3 Specialty Engineering'
THEN load('special.ckb') and
     do(Specialty) and
     remove_topic('Specialty') and
     do('System Management Menu').

IF ?System is '3.5.1.4 Manufacturing Engineering'
THEN load('Manufact.ckb') and
     do(Manufacturing) and
     remove_topic('Manufacturing') and
     do('System Management Menu').

IF ?System is '3.5.1.5 Security'
THEN load('Security.ckb') and
     do(Security) and
     remove_topic('Security') and
     do('System Management Menu').

IF ?System is '3.5.1.6 Communications'
THEN load('Commun.ckb') and
     do(Communications) and
     remove_topic('Communications') and
     do('System Management Menu').

IF ?System is HELP
THEN do(Help_System) and
     remove_topic('Help_System') and
     do('System Management Menu').

IF ?System is 'NONE OF THE ABOVE'
THEN exit().

end. (* System Management Menu *)

Topic Help_Management.
ASK('Select the area for which you desire additional
information

NOTE: The folling information is not written to disk.',
management, ['3.5.1 Systems Management', '3.5.2 Project
Management']).

Helptext(?management).
end. (* Help_Management *)

Topic Help_System.
ASK('Select the area for which you desire additional
information

NOTE: The following information is not written to disk.',
```

```
system, ['3.5.1.1 Design Engineering', '3.5.1.2 Logistics
Engineering','3.5.1.3 Specialty Engineering', '3.5.1.4
Manufacturing Engineering', '3.5.1.5 Security', '3.5.1.6
Communications']).

Helptext(?system).
end. (* Help_System *)

Topic Helptext(findstring).
Text is Read('Help.txt', ?findstring, '/end').
window(?findstring,,,,1,5,78,10).
Say(?text).
close_window().
end. (* Helptext *)

END. (* System/Project Management Menu *)
```

```
(*                          PMGMT.KB                                *)
(* PMGMT.KB OUTPUTS THE TEXT FOR SOW PARAGRAPH -                    *)
(* 3.5.2 PROJECT MANAGEMENT                                         *)


ASK('Input a file name.',outfile).
do('Program Management').

(* TOPIC PROGRAM MANAGEMENT IS THE PARENT TOPIC *)
Topic 'Program Management'.
write(?outfile,'3.5.2 Project Management',#n).
do(Delete_Data).

Topic Delete_Data.
(* Delete old data *)
EOF is number_to_char(26).
data is find('Pmgmt.dat').
IF ?data <> ?EOF
THEN dos('del Pmgmt.dat') and
     new_file('Pmgmt.dat') AND
     Close('Pmgmt.dat').
do(restore).
end. (* Delete_Data *)

Topic restore.
EOF is number_to_char(26).

data is find('Testeval.dat').
IF ?data <> ?EOF
THEN
Software_Test_Program is read('Testeval.dat',
'Software_Test_Program', '/') and
Formal_Qual_Review is read('Testeval.dat','FCA_OR_FQR','/')
ELSE Software_Test_Program is UNKNOWN.

data is find('Design.dat').
IF ?data <> ?EOF
THEN Design_Change is read('Design.dat','Design_Change','/')
ELSE
ASK('The system requires the design/redesign of:',
Design_Change, [Hardware,Software,Both, 'No design or
redesign']).

data is find('logistic.dat').
IF ?data <> ?EOF
THEN Warranty_Maintenance is read('logistic.dat',
'Warranty_Maintenance', '/')
and  NDI_ITEMS is read('logistic.dat','NDI_ITEMS','/')
ELSE NDI_ITEMS is UNKNOWN.

do('System Designed or Developed').
```

```
end. (* Restore *)


Topic Include_Paragraph(area,findstring).
Text is find('pmgmt.txt',,,?findstring,'/end').
write(?outfile,?text,#n).
window(?area,,,,1,7,78,13).
say(?text).
close_window().
close('pmgmt.txt').
end. (* Include_Paragraph *)

Topic 'System Designed or Developed'.
write([?outfile],'CONTRACT WORK BREAKDOWN STRUCTURE:',#n).
area is 'CONTRACT WORK BREAKDOWN STRUCTURE'.
IF ?Design_Change is Hardware OR
   ?Design_Change is Software OR
   ?Design_Change is Both
THEN 'System Designed or Developed' is YES
ELSE 'System Designed or Developed' is NO.

IF ?'System Designed or Developed' is YES
THEN Include_Paragraph(?area,'System_Designed_or_Developed
-YES')
and  do(Contract_Type)
Else do(Unique_Sched_Management).
end. (*'System Designed or Developed'*)

Topic Contract_Type.
ASK (' This contract is:', Contract_Type,
[ Firm_Fixed_Price, Incentive, Unknown]).

IF ?Contract_Type is Incentive
THEN do(Dollars_FSD_Contract)
ELSE do(Unique_Sched_Management).
end. (*Contract_Type*)

Topic Dollars_FSD_Contract.
ASK ('The dollar value for this effort is:
A. $40 Million Dollars and Above
B. Between $2-$40 Million Dollars
C. Less than $2 Million Dollars', Dollars_FSD_Contract,[ A, B,
C]).

IF ?Dollars_FSD_Contract is A
THEN Include_Paragraph(?area,'Dollars_FSD_Contract_CWBS
      -A_or_B')
and  area is 'Contract Schedule and Control System'
and  WRITE(?outfile, 'Contract Schedule and Control System:',
     #n)
and  Include_Paragraph(?area,'Dollars_FSD_Contract_CSCS-A')
```

```
and  area is 'Cost Information System'
and  WRITE(?outfile, 'Cost Information System:',#n)
and  Include_Paragraph(?area,'Dollars_FSD_Contract_Cost
     -A_or_B').

IF ?Dollars_FSD_Contract is B
THEN Include_Paragraph(?area,'Dollars_FSD_Contract_CWBS
     -A_or_B')
and  area is 'Contract Schedule and Cost System'
and  WRITE(?outfile, 'Contract Schedule and Cost System:',#n)
and  Include_Paragraph(?area,'Dollars_FSD_Contract_CSCS-B')
and  area is 'Cost Information System'
and  WRITE(?outfile, 'Cost Information System:',#n) and

Include_Paragraph(?area,'Dollars_FSD_Contract_Cost-A_or_B').

do(Unique_Sched_Management).
end. (*Dollars_FSD_Contract*)

Topic Unique_Sched_Management.
write([?outfile],'SCHEDULE MANAGEMENT:').
area is 'SCHEDULE MANAGEMENT'.
ASK ('Does this program require a unique method of
schedule management?', Unique_Sched_Management,[ YES, NO,
UNKNOWN]).

IF ?Unique_Sched_Management is YES
THEN Include_Paragraph(?area,'Unique_Sched_Management-YES').

IF ?Unique_Sched_Management is NO
THEN Include_Paragraph(?area,'Unique_Sched_Management-NO').
do(Config_Required).
end. (*Unique_Sched_Management*)

Topic Config_Required.
area is 'CONFIGURATION MANAGEMENT'.
Include_Paragraph(?area,'-STINFO').
write([?outfile],'CONFIGURATION MANAGEMENT:',#n).

ASK ('Is Configuration Management applicable to this
program?', Config_Required,[YES,NO,UNKNOWN]).

IF ?Config_Required is NO
THEN Include_Paragraph(?area,'Config_Required-NO') and
     do(Contractor_Config).

IF ?Config_Required is YES or
   ?Config_Required is UNKNOWN
THEN do(Contractor_Config).
end. (*Config_Required*)
```

```
Topic Contractor_Config.
IF ?Config_Required is NO
THEN Contractor_Config is UNKNOWN
ELSE
ASK ('The contractors Configuration Management system has
been:

A.  Previously validated by the government and no new
    additional configuration management requirements have been
    invoked.
B.  Previously validated by the government however new or
    additional configuration management requirements have been
    invoked
C.  Has not been previously validated by the government
D.  Unknown', Contractor_Config,[ A, B, C, D]).

IF ?Contractor_Config is A OR
   ?Contractor_Config is B
THEN  Include_Paragraph(?area,'Contractor_Config-A_or_B').
do(Interface_Control).
end. (*Contractor_Config*)

Topic Interface_Control.
IF ?Config_Required is NO
THEN Interface_Control is UNKNOWN
ELSE
ASK ('The contractor will:

A.  Establish and chair an Interface Control Working Group
    (ICWG).
B.  Be a member of an Interface Control Working Group (ICWG).
C.  Have other interface control responsibilities.  The ICWG
    is not applicable.
D.  Interface control is not required
E.  Unknown', Interface_Control,[ A, B, C, D, E]).


IF ?Interface_Control is A
THEN Include_Paragraph(?area,'Interface_Control-A').

IF ?Interface_Control is B
THEN Include_Paragraph(?area,'Interface_Control-B').

IF ?Interface_Control is C
THEN Include_Paragraph(?area,'Interface_Control-C').

IF ?Interface_Control is D
THEN Include_Paragraph(?area,'Interface_Control-D').
do(Type_Baseline).
end. (*Interface_Control*)
```

```
Topic Type_Baseline.
IF ?Config_Required is NO
THEN Type_Baseline is UNKNOWN
ELSE
ASK ('The contractor is responsible for the:

A. Functional Configuration Identification and/or baseline.
B. Allocated Configuration Identification and/or baseline.
C. Product Configuration Identification and/or baseline.
D. Unknown', Type_Baseline,[ A, B, C, D]).

IF ?Type_Baseline is A
THEN Include_Paragraph(?area,'Type_Baseline-A').

IF ?Type_Baseline is B
THEN Include_Paragraph(?area,'Type_Baseline-B').

IF ?Type_Baseline is C
THEN Include_Paragraph(?area,'Type_Baseline-C').
do(ECPS).
end. (*Type_Baseline*)

Topic ECPS.
IF ?Config_Required is NO
THEN ECPS is UNKNOWN
ELSE
ASK ('Will engineering drawings and specifications be
prepared for hardware and software configuration items that
are to be developed, procured and/or delivered?'
, ECPS,[ YES, NO, UNKNOWN]).

IF ?ECPS is YES
THEN Include_Paragraph(?area,'ECPS-YES').
do(Count_Config_Items).
end. (*ECPS*)

Topic Count_Config_Items.
IF ?Config_Required is NO
THEN Count_Config_Items is UNKNOWN
ELSE
ASK ('Will the contractor be required to account for and
document configuration items at site locations?',
Count_Config_Items,[ YES, NO, UNKNOWN]).

IF ?Count_Config_Items is YES
THEN Include_Paragraph(?area,'Count_Config_Items-YES').
do(Status_Acct_Reports).
end. (*Count_Config_Items*)

Topic Status_Acct_Reports.
IF ?Config_Required is NO
```

```
THEN Status_Acct_Reports is UNKNOWN
ELSE
ASK ('Will Status Accounting Reports be required in addition
to the Configuration Item Development Records?',
Status_Acct_Reports, [ YES, NO, UNKNOWN]).

IF ?Status_Acct_Reports is YES
THEN Include_Paragraph(?area,'Status_Acct_Reports-YES').
do(System_Des_Or_Mod).
end. (*Status_Acct_Reports*)

Topic System_Des_Or_Mod.
IF ?Design_Change is Hardware OR
   ?Design_Change is Software OR
   ?Design_Change is UNKNOWN
THEN System_Des_Or_Mod is YES
ELSE
IF ?Config_Required is NO
THEN System_Des_Or_Mod is UNKNOWN
ELSE
ASK ('Will the system contain hardware/software of, new
designs, modifications of existing designs or new sources?',
System_Des_Or_Mod,[YES,NO,UNKNOWN]).

IF ?System_Des_Or_Mod is YES
THEN Include_Paragraph(?area,'System_Des_Or_Mod-YES').
do(Dev_Contractor_Is_Prod).
end. (*System_Des_or_Mod*)

Topic Dev_Contractor_Is_Prod.
IF ?Config_Required is NO
THEN Dev_Contractor_Is_Prod is UNKNOWN
ELSE
ASK ('Will the development contractor be the production
contractor?', Dev_Contractor_Is_Prod,[ YES, NO, UNKNOWN]).

IF ?Dev_Contractor_Is_Prod is YES
THEN Include_Paragraph(?area,'Dev_Contractor_Is_Prod-YES').

IF ?Dev_Contractor_Is_Prod is NO
THEN Include_Paragraph(?area,'Dev_Contractor_Is_Prod-NO').
do(Formal_Qual_Review).
end. (*Dev_Contractor_Is_Prod*)

Topic Formal_Qual_Review.
IF ?Config_Required is NO
THEN Formal_Qual_Review is UNKNOWN
ELSE
    IF exists(Formal_Qual_Review)
    THEN Formal_Qual_Review is Formal_Qual_Review
    ELSE
```

```
    ASK ('Do you intend to conduct a Formal Qualification
    Review (FQR)?', Formal_Qual_Review,[ YES, NO, UNKNOWN]).

IF ?Formal_Qual_Review is YES
THEN Include_Paragraph(?area,'Formal_Qual_Review-YES').
do(Nomen_required).
end. (*Formal_Qual_Review*)

Topic Nomen_required.
write([?outfile],'NOMENCLATURE:',#n).
area is 'NOMENCLATURE'.
IF ?Config_Required is YES OR
   ?System_Des_Or_Mod is YES OR
   ?ECPS is YES OR
   ?Design_Change is Software OR
   ?Design_Change is Hardware OR
   ?Design_Change is Both OR
   ?'System Designed or Developed' is YES
THEN Nomen_required is YES
ELSE
ASK ('Will this program require Nomenclature?',
Nomen_required,[ YES, NO, UNKNOWN]).

IF ?Nomen_required is YES
THEN Include_Paragraph(?area,'Nomen_required-YES').
do(Design).
end. (*Nomen_required*)

Topic Design.
write([?outfile],'MANUFACTURING MANAGEMENT:',#n).
area is 'MANUFACTURING MANAGEMENT'.
Design is Design_Change.
IF ?Design is Hardware
THEN Include_Paragraph(?area,'Design_Change-Hardware').

IF ?Design is Software
THEN Include_Paragraph(?area,'Design_Change-Software').

IF ?Design is Both
THEN Include_Paragraph(?area,'Design_Change-Hardware') and
     Include_Paragraph(?area,'Design_Change-Software').

do(Funding_Cap).
end. (*Design*)

Topic Funding_Cap.
ASK ('Will FSD funding exceed $100 Million,
or will Production funding exceed $20 million annually
or $100 million cumulatively?', Funding_Cap,[ YES, NO,
UNKNOWN]).
```

```
IF ?Funding_Cap is YES
THEN Include_Paragraph(?area,'Funding_Cap-YES').
do(Type_Prod_Contract).
end. (*Funding_Cap*)

Topic Type_Prod_Contract.
ASK ('The follow-on Production Contract will be',
Type_Prod_Contract,[ Competitive, Sole_Source]).

IF ?Type_Prod_Contract is Competitive
THEN Include_Paragraph(?area,'Type_Prod_Contract
     -Competitive').

IF ?Type_Prod_Contract is Sole_Source
THEN
Include_Paragraph(?area,'Type_Prod_Contract-Sole_Source').

close('Pmgmt.dat').
WRITE('Pmgmt.dat',#o,'Type_Prod_Contract'.?Type_Prod_Contract.
'/').

do(Dollars_Prod_Contract).
end. (*Type_Prod_Contract*)

Topic Dollars_Prod_Contract.
ASK (' The dollar value of the follow-on Production Contract
will be:

A. $100 million and above
B. $50 - 99 million
C. $0 - 49 million
D. Unknown', Dollars_Prod_Contract,[ A, B, C, D]).

IF ?Dollars_Prod_Contract is A OR
   ?Dollars_Prod_Contract is D
THEN Include_Paragraph(?area,'Dollars_Prod_Contract-A_or_D').

IF ?Dollars_Prod_Contract is B
THEN Include_Paragraph(?area,'Dollars_Prod_Contract-B').

IF ?Dollars_Prod_Contract is C
THEN Include_Paragraph(?area,'Dollars_Prod_Contract-C').
do(Computer_Resources).
end. (*Dollars_Prod_Contract*)

Topic Computer_Resources.
write([?outfile],'COMPUTER RESOURCES:',#n).
area is 'COMPUTER RESOURCES'.
IF ?Software_Test_Program is YES OR
   ?Design_Change is Software OR
   ?Design_Change is Both
```

```
THEN Computer_Resources is YES
ELSE
ASK ('Does the system/project involve computer resources?',
Computer_Resources,[ YES, NO, UNKNOWN]).

IF ?Computer_Resources is YES
THEN Include_Paragraph(?area,'Computer_Resources-YES').

IF ?Computer_Resources is NO
THEN Include_Paragraph(?area,'Computer_Resources-NO').
do(NDI_USED).
end. (*Computer_Resources*)

Topic NDI_USED.
IF ?NDI_ITEMS is A OR
   ?NDI_ITEMS is B
THEN NDI_USED is YES
ELSE
ASK (' In the RFP, will you provide the contractor the
option to use Off-The-Shelf software?', NDI_USED,[ YES, NO,
UNKNOWN]).

IF ?NDI_USED is YES
THEN Include_Paragraph(?area,'NDI_USED-YES').
do(How_Travel).
end. (*NDI_USED*)

Topic How_Travel.
write([?outfile],'TRAVEL:',#n).
area is 'TRAVEL'.
ASK('During Full-Scale Development, the contractor will:
A.  TRAVEL VIA GOVERNMENT EXPENSE ONLY
B.  TRAVEL VIA CONTRACTOR EXPENSE ONLY
C.  TRAVEL VIA BOTH GOVERNMENT AND CONTRACTOR EXPENSE
D.  NOT TRAVEL.
E.  UNKNOWN.',
How_Travel, [A,B,C,D,E]).

IF ?How_Travel is A
THEN Include_Paragraph(?area,'How_Travel-A') and
     do(Travel_To).

IF ?How_Travel is B
THEN Include_Paragraph(?area,'How_Travel-B').

IF ?How_Travel is C
THEN Include_Paragraph(?area,'How_Travel-C') and
     do(Travel_To).

IF ?How_Travel is D
THEN Include_Paragraph(?area,'How_Travel-D').
```

```
IF ?How_Travel is E
THEN do(Travel_To).
end. (* How_Travel*)

Topic Travel_To.
ASK(' The contractor will travel at government expense to and
from:

A. THE 48 CONTIGUOUS STATES
B. ALASKA AND HAWAII
C. CONUS AND OVERSEAS AREAS
D. ALL THE ABOVE
E. THE 48 CONTIGUOUS STATES & ALASKA AND HAWAII
F. THE 48 CONTIGUOUS STATES & CONUS AND OVERSEAS AREAS.
G. ALASKA AND HAWAII & CONUS AND OVERSEAS AREAS.
H. UNKNOWN',
Travel_To, [A,B,C,D,E,F,G,H]).

IF ?Travel_To is A
THEN Include_Paragraph(?area,'Travel_To-A').

IF ?Travel_To is B
THEN Include_Paragraph(?area,'Travel_To-B').

IF ?Travel_To is C
THEN Include_Paragraph(?area,'Travel_To-C').

IF ?Travel_To is D
THEN Include_Paragraph(?area,'Travel_To-A') and
     Include_Paragraph(?area,'Travel_To-B') and
     Include_Paragraph(?area,'Travel_To_C').

IF ?Travel_To is E
THEN Include_Paragraph(?area,'Travel_To-A') and
     Include_Paragraph(?area,'Travel_To-C').

IF ?Travel_To is F
THEN Include_Paragraph(?area,'Travel_To-A') and
     Include_Paragraph(?area,'Travel_To-B').

IF ?Travel_To is G
THEN Include_Paragraph(?area,'Travel_To-B') and
     Include_Paragraph(?area,'Travel_To-C').

end. (* Travel_To *)

End. (* program management *)
```

```
(*                    SECURITY.KB                              *)
(* SECURITY.KB PRODUCES THE TEXT FOR SOW PARAGRAPH -  *)
(* 3.5.1.5 SECURITY.                                          *)

ASK('input a file name',outfile).
do(Security).

(* TOPIC SECURITY IS THE PARENT TOPIC (MAIN TOPIC)    *)
Topic Security.
write(?outfile,'3.5.1.5 Security',#n).

do(Security_Thru_Documentation).

Topic Include_Paragraph(area,findstring).
Text is find('security.txt',,,?findstring,'/end').
window(?area,,,,1,7,78,13).
SAY(?text).
WRITE(?outfile,?text,#n).
close_window().
close('security.txt').
end. (* Include_Paragraph *)

Topic Security_Thru_Documentation.
write(?outfile,'GENERAL SECURITY:',#n).
area is 'GENERAL SECURITY'.
ASK ('Have security requirements been identified through
program direction or other valid threat documentation?',
Security_Thru_Documentation, [YES, NO, UNKNOWN]).

IF ?Security_Thru_Documentation is YES
THEN
Include_Paragraph(?area,'Security_Thru_Documentation-YES').
do(Security_Required).
End. (* Security_Thru_Documentation *)

Topic Security_Required.
write(?outfile,'COMMUNICATIONS SECURITY/TEMPEST.',#n).
area is 'COMMUNICATIONS SECURITY/TEMPEST'.
ASK ('Will the equipment generate, handle, process, or use
classified or national security related information?',
Security_Required, [YES, NO, UNKNOWN]).

IF ?Security_Required is YES
THEN Include_Paragraph(?area,'Security_Required-YES') and
     do(Who_Tempest_Test).

IF ?Security_Required is UNKNOWN
THEN Include_Paragraph(?area,'Security_Required-UNKNOWN') and
     do(Who_Tempest_Test).
End. (* Security_Required *)
```

```
Topic Who_Tempest_Test.
ASK ('The system or equipment will be:

A. Tempest tested by the contractor or subcontractor.
   (Note: A contractor will perform testing only if
   Air Force testing resources are not available.)
B. Tempest Tested by the Government
C. Has been previously tempest tested.
D. Unknown.', Who_Tempest_Test, [A, B, C, D]).

IF ?Who_Tempest_Test is A
THEN Include_Paragraph(?area,'Who_Tempest_Test-A').

IF ?Who_Tempest_Test is B
THEN Include_Paragraph(?area,'Who_Tempest_Test-B').

IF ?Who_Tempest_Test is C
THEN Include_Paragraph(?area,'Who_Tempest_Test-C').

IF ?Who_Tempest_Test is D
THEN Include_Paragraph(?area,'Who_Tempest_Test-UNKNOWN').
do(Contractor_Install).
End. (*Who_Tempest_Test*)

Topic Contractor_Install.
ASK ( 'Will the contractor install the equipment?',
Contractor_Install,[YES, NO, UNKNOWN]).

IF ?Contractor_Install is YES
THEN Include_Paragraph(?area,'Contractor_Install-YES').

IF ?Contractor_Install is NO
THEN Include_Paragraph(?area,'Contractor_Install-NO').
do(Secure_Comm_Required).
End. (* Contractor_Install *)

Topic Secure_Comm_Required.
ASK ('Will classified information be transmitted over
radio systems, telephone circuits, or intercom systems?

(Note: This relates to both the system/equipment under
development/procurment and classified information processed at
the contractor facility.)', Secure_Comm_Required,
[YES, NO, UNKNOWN]).

IF ?Secure_Comm_Required is YES
THEN Include_Paragraph(?area,'Secure_Comm_Required-YES') and
     do(Type_Comsec_Material).

IF ?Secure_Comm_Required is UNKNOWN
THEN Include_Paragraph(?area,'Secure_Comm_Required-UNKNOWN')
```

```
and do(Type_Comsec_Material).
End. (* Secure_Comm_Required *)


Topic Type_Comsec_Material.
ASK ('The contract requires:

A.  Access to or storage of classified comsec material.
    (Examples of classified COMSEC material include COMSEC
    installation standards, COMSEC design information,
    National COMSEC/EMSEC information memorandums (NACSEM),
    COMSEC equipment engineering bulletin COMSEC engineering
    systems documents (CSESD))
B.  Access to or storage of accountable COMSEC
    material/equipment.
    (This includes all COMSEC equipment/material shipped,
    received, or safeguarded within the COMSEC material
    control system.  Some examples include crypto maintenance
    and operational manuals, crypto equipment, all keying
    material, and ancillary equipment (Key Fill Devices, PC
    Boards, ETC.))
C.  Both of the above
D.  None of the above.  E.  Unknown',
Type_Comsec_Material,[A, B, C, D, E]).

IF ?Type_Comsec_Material is A
THEN Include_Paragraph(?area,'Type_Comsec_Material-A').

IF ?Type_Comsec_Material is B
THEN Include_Paragraph(?area,'Type_Comsec_Material-B').

IF ?Type_Comsec_Material is C
THEN Include_Paragraph(?area,'Type_Comsec_Material-A') and
     Include_Paragraph(?area,'Type_Comsec_Material-B').

IF ?Type_Comsec_Material is E
THEN Include_Paragraph(?area,'Type_Comsec_Material-UNKNOWN').
End. (* Type_Comsec_Mat *)

END. (* Security *)
```

```
(*                    SPARES.KB                              *)
(* SPARES.KB PRODUCES THE TEXT FOR SOW PARAGRAPH -          *)
(* 3.10 INITIAL SPARES AND REPAIR PARTS                     *)

ASK('Input a file name',outfile).
do('Initial Spare/Repair Parts').

(* TOPIC INITIAL SPARE/REPAIR PARTS IS THE PARENT TOPIC  *)
Topic 'Initial Spare/Repair Parts'.
write([?outfile],'3.10  Initial Spares and Repair Parts').
area is 'INITIAL SPARES/REPAIR PARTS'.
do(Hardware_Supported_By).

Topic Include_Paragraph(area,findstring).
Text is find('SPARES.TXT',,,?findstring,'/end').
window(?area,,,,1,7,78,13).
say(?text).
write([?outfile],?text,#n).
close_window().
close('SPARES.TXT').
end. (* Include_Paragraph *)

Topic Hardware_Supported_By.
ASK('The hardware provided in this phase will be deployed and
supported by:
A. AFLC
B. OTHER AIR FORCE COMMANDS
C. NO HARDWARE IS PROVIDED
D. UNKNOWN',
Hardware_Supported_By, [A, B,C,D]).

IF ?Hardware_Supported_By is A
THEN Include_Paragraph(?area,'Hardware_Supported_By-A') and
     do(SAIP).

IF ?Hardware_Supported_By is C
THEN Include_Paragraph(?area,'Hardware_Supported_By-C').

IF ?Hardware_Supported_By is D
THEN Include_Paragraph(?area,'Hardware_Supported_By-D') and
     do(SAIP).

end. (*Hardware_Supported_By*)

Topic SAIP.
ASK('Has Spares Acquisition Integrated with Production (SAIP)
been selected?

NOTE:  SAIP should be beneficial to the Government for spares
with a stable design.  SAIP allows the Government to buy
spares at a lower unit cost when it can be integrated with
```

```
installation quantities.  Unit prices under SAIP should be
lower. SAIP should be considered as an economic method of
provisioning!', SAIP, [YES, NO, UNKNOWN]).

IF ?SAIP is YES
THEN Include_Paragraph(?area,'SAIP-YES').
do(Provisioning_Guidance_Conference).
end. (* SAIP *)

Topic Provisioning_Guidance_Conference.
ASK('A Provisioning Guidance Conference will:',
Provisioning_Guidance_Conference, ['Be Called', 'Not Be
Called', UNKNOWN]).

IF ?Provisioning_Guidance_Conference is 'Be Called'
THEN Include_Paragraph(?area,'Provisioning_Guidance_Conference
-Be_Called').

IF ?Provisioning_Guidance_Conference is UNKNOWN
THEN Include_Paragraph(?area,'Provisioning_Guidance_Conference
-UNKNOWN').
end. (* Provisioning_Guidance_Conference *)

end. (* Initial Spare/Repair Parts *)
```

```
(*                          SPECIAL.KB                                  *)
(* SPECIAL.KB PRODUCES THE TEXT FOR SOW PARAGRAPH -                     *)
(* 3.5.1.3 SPECIALTY ENGINEERING                                        *)


ASK('Input a file name:',outfile).
do(Specialty).

(* TOPIC SPECIALTY IS THE PARENT TOPIC                                 *)
Topic Specialty.
write([?outfile],'3.5.1.3 Specialty Engineering',#n).
do(Restore).

Topic Restore.
EOF is number_to_char(26).
data is find('Design.dat').
IF ?Data <> ?EOF
THEN Prime_Mission_Equipment is read('Design.dat',
'Prime_Mission_Equipment', '/')
ELSE
ASK('This contract requires:

A. PRIME MISSION EQUIPMENT FOR A RADAR, COMMUNICATIONS, OR
   NAVIGATIONAL SYSTEM THAT USES THE ELECTROMAGNETIC SPECTRUM.
B. PRIME MISSION EQUIPMENT NOT FOR A RADAR, COMMUNICATIONS, OR
   NAVIGATIONAL SYSTEM THAT USES THE ELECTROMAGNETIC SPECTRUM.
C. NO PRIME MISSION EQUIPMENT.
D. UNKNOWN', Prime_Mission_Equipment, [A,B,C,D]).

data is find('logistic.dat').
IF ?data <> ?EOF
THEN NDI_ITEMS is read('logistic.dat','NDI_ITEMS','/')
ELSE
ASK ( 'The system will have:

                A. Some Non Developmental Items
                B. All Non Developmental Items
                C. No Non Developmental Items
                D. UNKNOWN',
NDI_ITEMS,[ A,B,C,D]).
do(System_Safety_Manager).
end. (* Restore *)


TOPIC System_Safety_Manager.
write([?outfile],'SYSTEM SAFETY:',#n).
area is 'SYSTEM SAFETY'.

ASK('Do you know the System Safety Manager (SSM) for your
program office?', System_Safety_Manager, [YES,NO, UNKNOWN]).
```

```
IF ?System_Safety_Manager is YES
THEN Include_Paragraph(?area,'System_Safety_Manager-YES')
ELSE Include_Paragraph(?area,'System_Safety_Manager-NO').
Do(NDI).
END. (* System_Safety_Manager *)

TOPIC NDI.
IF ?NDI_ITEMS is A or
   ?NDI_ITEMS is B
THEN Include_Paragraph(?area,'NDI_ITEMS-A_or_B') and
     Do(GFE_Required).

IF ?NDI_ITEMS is C
THEN Do(GFE_Required).

IF ?NDI_ITEMS is D
THEN Include_Paragraph(?area,'NDI_ITEMS-D') and
     Do(GFE_Required).
End. (* NDI *)

TOPIC GFE_Required.
ASK ( 'Does this program require GFE?',GFE_Required,
[ YES,NO,UNKNOWN]).

IF ?GFE_Required is YES
THEN Include_Paragraph(?area,'GFE_Required-YES')

ELSE
   IF ?GFE_Required is NO
   THEN Include_Paragraph(?area,'GFE_Required-NO')
   ELSE Include_Paragraph(?area,'GFE_Required-UNKNOWN').
Do(Safety_Safe_Design).
End. (* GFE_Required *)

TOPIC Safety_Safe_Design.
ASK ('Will the system comply with applicable safety
(safety related) military, federal (public law),national and
industrial codes or standards to ensure a safe design?',
Safety_Safe_Design,[YES, NO, UNKNOWN]).

IF ?Safety_Safe_Design is YES
THEN Include_Paragraph(?area,'Safety_Safe_Design-YES')
ELSE
     IF ?Safety_Safe_Design is NO
     THEN Include_Paragraph(?area,'Safety_Safe_Design-NO').
Do(SSPP_Written_By).
End. (* Safety_Safe_Design *)

TOPIC SSPP_Written_By.
ASK ( 'The System Safety Plan (SSPP) is to be written by the',
SSPP_Written_By,[ Contractor, SPO, UNKNOWN]).
```

```
IF ?SSPP_Written_By is Contractor
THEN Include_Paragraph(?area,'SSPP_Written_By-Contractor')
ELSE
    IF ?SSPP_Written_By is SPO
    THEN Include_Paragraph(?area,'SSPP_Written_By-SPO')
    ELSE Include_Paragraph(?area,'SSPP_Written_By-UNKNOWN').
Do(Contractor_as_Integrator).
End. (* SSPP_Written_By *)

TOPIC Contractor_as_Integrator.
ASK ('Will there be a contractor designated as integrator with
management surveillance of associated contractors,
subcontractor, or architect and engineering firms?',
Contractor_as_Integrator, [YES, NO, UNKNOWN]).

IF ?Contractor_as_Integrator is YES
THEN Include_Paragraph(?area,'Contractor_as_Integrator-YES')
ELSE
    IF ?Contractor_as_Integrator is UNKNOWN
    THEN
    Include_Paragraph(?area,'Contractor_as_Integrator
    -UNKNOWN').
Do(Safety_At_Reviews).
END. (* Contractor_as_Integrator *)

TOPIC Safety_At_Reviews.
ASK ( 'Do you want the contractor to discuss
System Safety at management and/or design reviews?',
Safety_At_Reviews, [ Yes, No, Unknown]).

IF ?Safety_At_Reviews is YES
THEN Include_Paragraph(?area,'Safety_At_Reviews-YES')
ELSE
     IF ?Safety_At_Reviews is NO
     THEN Include_Paragraph(?area,'Safety_At_Reviews-NO')
     ELSE
     Include_Paragraph(?area,'Safety_At_Reviews-UNKNOWN').

Do(SSG_Or_SSWG).
End. (* Safety_At_Reviews *)

TOPIC SSG_Or_SSWG.
ASK ( ' Is a System Safety Group (SSG) or System Safety
Working Group (SSWG) being established for this acquisition
program?', SSG_Or_SSWG,[ YES, No, Unknown]).

IF ?SSG_Or_SSWG is YES
THEN Include_Paragraph(?area,'SSG_Or_SSWG-YES').

IF ?SSG_Or_SSWG is NO or
```

```
   ?SSG_Or_SSWG is Unknown
THEN Include_Paragraph(?area,'SSG_Or_SSWG-NO_or_UNKNOWN').
Do(Close_Loop_Track).
End. (* SSG_Or_SSWG *)

TOPIC Close_Loop_Track.
ASK ( 'Will the contractor be required to close-loop
track identified hazards until eliminated or controlled?',
Close_Loop_Track,[ YES, NO, UNKNOWN]).

IF ?Close_Loop_Track is YES
THEN Include_Paragraph(?area,'Close_Loop_Track-YES')
ELSE
     IF ?Close_Loop_Track is NO
     THEN Include_Paragraph(?area,'Close_Loop_Track-NO')
     ELSE Include_Paragraph(?area,'Close_Loop_Track-UNKNOWN').

Do(Periodic_Reports).
End. (* Close_Loop_Track *)

TOPIC Periodic_Reports.
ASK ( 'Are periodic reports of the System Safety Management
and Engineering activities required during the contract
period?', Periodic_Reports,[ YES, NO, UNKNOWN]).

IF ?Periodic_Reports is YES
THEN Include_Paragraph(?area,'Periodic_Reports-YES')
ELSE
     IF ?Periodic_Reports is NO
     THEN Include_Paragraph(?area,'Periodic_Reports-NO')
     ELSE Include_Paragraph(?area,'Periodic_Reports-UNKNOWN').

Do(Prelim_Hazard).
End. (*Periodic_Reports*)

TOPIC Prelim_Hazard.
ASK ( 'The following will be required:

                    A. A Preliminary Hazard List
                    B. A Preliminary Hazard Analysis
                    C. Both of the Above
                    D. None of the Above
                    E. Unknown',
 Prelim_Hazard,[ A, B, C, D, E]).

IF ?Prelim_Hazard is A
THEN Include_Paragraph(?area,'Prelim_Hazard-A').

IF ?Prelim_Hazard is B
THEN Include_Paragraph(?area,'Prelim_Hazard-B').
```

```
IF ?Prelim_Hazard is C
THEN Include_Paragraph(?area,'Prelim_Hazard-A') and
     Include_Paragraph(?area,'Prelim_Hazard-B').

IF ?Prelim_Hazard is D OR
   ?Prelim_Hazard is E
THEN Include_Paragraph(?area,'Prelim_Hazard-D_or_E').
Do(Analysis_Required).
End. (*Prelim_Hazard *)

TOPIC Analysis_Required.
ASK ( 'The following analyses will be required:

                          A. Subsystem Hazard Analysis
                          B. System Hazard Analysis
                          C. Both of the Above
                          D. None of the Above
                          E. Unknown',
 Analysis_Required,[ A, B, C, D, E]).

IF ?Analysis_Required is A
THEN Include_Paragraph(?area,'Analysis_Required-A').

IF ?Analysis_Required is B
THEN Include_Paragraph(?area,'Analysis_Required-B').

IF ?Analysis_Required is C
THEN Include_Paragraph(?area,'Analysis_Required-A') and
     Include_Paragraph(?area,'Analysis_Required-B').

IF ?Analysis_Required is D
THEN Include_Paragraph(?area,'Analysis_Required-D').

IF ?Analysis_Required is E
THEN Include_Paragraph(?area,'Analysis_Required-E').
Do(Safety_Test).
End. (* Analysis_Required *)

TOPIC Safety_Test.
ASK ( 'Will the system have safety critical hardware,
software, or procedures which would require verification
by analysis, inspection, demonstration or test?',
Safety_Test,[ YES, NO, UNKNOWN]).

IF ?Safety_Test is YES
THEN Include_Paragraph(?area,'Safety_Test-YES')
ELSE
     IF ?Safety_Test is NO
     THEN Include_Paragraph(?area,'Safety_Test-NO')
     ELSE Include_Paragraph(?area,'Safety_Test-UNKNOWN').
```

```
Do(Safety_Training).
End. (* Safety_Test *)

TOPIC Safety_Training.
ASK ( 'Will the system under development involve safety
hazards requiring contractor design personnel and managers,
and/or contractor government test personnel to receive
specialized safety training?', Safety_Training,[ YES, NO,
UNKNOWN]).

IF ?Safety_Training is YES
THEN Include_Paragraph(?area,'Safety_Training-YES')
ELSE
     IF ?Safety_Training is UNKNOWN
     THEN Include_Paragraph(?area,'Safety_Training-UNKNOWN').
Do(Mishap_Risk).
End. (* Safety_Training *)

TOPIC Mishap_Risk.
ASK ( 'Does the system require a comprehensive evaluation
of the mishap risk being assumed (by the Air Force)
prior to test, operation of a system, or contract
completion?', Mishap_Risk,[ YES, NO, UNKNOWN]).

IF ?Mishap_Risk is YES
THEN Include_Paragraph(?area,'Mishap_Risk-YES')
ELSE
     IF ?Mishap_Risk is NO
     THEN Include_Paragraph(?area,'Mishap_Risk-NO')
     ELSE Include_Paragraph(?area,'Mishap_Risk-UNKNOWN').

Do(Analyze_ECPS).
End. (* Mishap_Risk *)

TOPIC Analyze_ECPS.
ASK ( 'Do you want the contractor to perform a safety
analysis of an Engineering Change Proposal (ECP) (DI-E-3128/T)
or a request for deviation/waiver (DI-E-3129/T) for this
acquisition', Analyze_ECPS,[ YES, NO, UNKNOWN]).

IF ?Analyze_ECPS is YES
THEN Include_Paragraph(?area,'Analyze_ECPS-YES').

IF ?Analyze_ECPS is UNKNOWN
THEN Include_Paragraph(?area,'Analyze_ECPS-UNKNOWN').

Do(Software_Analysis).
End. (* Analyze_ECPS *)

TOPIC Software_Analysis.
ASK ( ' Will there be any critical software controlled
```

```
operator information, or command and control functions
associated with The system?', Software_Analysis,[ YES, NO,
UNKNOWN]).

IF ?Software_Analysis is YES
THEN Include_Paragraph(?area,'Software_Analysis-YES').

IF ?Software_Analysis is UNKNOWN
THEN Include_Paragraph(?area,'Software_Analysis-UNKNOWN').

Do(Be_In_Weather).
End. (* Software_Analysis *)


TOPIC Be_In_Weather.
write([?outfile],'AEROSPACE METEORLOGICAL ENVIRONMENT:',#n).
area is 'AEROSPACE METEORLOGICAL ENVIRONMENT'.
ASK ( 'The system will require operation, non-operation,
transport, and/or storage or components and subsystems:

A.  UNDER A RANGE OF METEOROLOGICAL (WEATHER) CONDITIONS IN
    THE EARTHS ATMOSPHERE
B.  UNDER A RANGE OF CONDITIONS IN SPACE
C.  BOTH IN THE EARTHS ATMOSPHERE AND IN SPACE
D.  NEITHER IN THE EARTHS ATMOSPHERE NOR IN SPACE
E.  UNKNOWN',
Be_In_Weather,[ A, B, C, D, E]).

IF ?Be_In_Weather is A
THEN Include_Paragraph(?area,'Be_In_Weather-A') and
     Do(Use_Weather_Data).

IF ?Be_In_Weather is B
THEN Include_Paragraph(?area,'Be_In_Weather-B') and
     do(Use_Weather_Data).

IF ?Be_In_Weather is C
THEN Include_Paragraph(?area,'Be_In_Weather-A') AND
     Include_Paragraph(?area,'Be_In_Weather-B') and
     do(Use_Weather_Data).

IF ?Be_In_Weather is D
THEN do(Electromagnetic_Radiation).

IF ?Be_In_Weather is E
THEN do(Use_Weather_Data).
End. (* Be_In_Weather *)

TOPIC Use_Weather_Data.
ASK ( 'Is the system required to transmit, receive, store
or process weather data?', Use_Weather_Data,[ YES, NO,
```

```
UNKNOWN]).

IF ?Use_Weather_Data is YES
THEN Include_Paragraph(?area,'Use_Weather_Data-YES').

Do(Electromagnetic_Radiation).
End. (* Use_Weather_Data *)

TOPIC Electromagnetic_Radiation.
IF ?Prime_Mission_Equipment is YES
THEN Electromagnetic_Radiation is YES
ELSE
ASK ( 'Does the system emit and/or receive electromagnetic
radiation?', Electromagnetic_Radiation,[ YES, NO, UNKNOWN]).

IF ?Electromagnetic_Radiation is YES
THEN Include_Paragraph(?area,'Electromagnetic_Radiation-YES').
Do(Lessons_Learned).
End. (* Electromagnetic_Radiation *)

TOPIC Lessons_Learned.
ASK ( 'Does this program/project have a lessons
learned section?', Lessons_Learned,[ YES, NO, UNKNOWN]).

IF ?Lessons_Learned is YES or
   ?Lessons_Learned is UNKNOWN
THEN
Include_Paragraph(?area,'Lessons_Learned-YES_or_UNKNOWN').
Do(Delivery_Terms_FOB).
End. (* Lessons_Learned *)

TOPIC Delivery_Terms_FOB.
write([?outfile],'PRESERVATION, PACKAGING, AND PACKING:',#n).
area is 'PRESERVATION, PACKAGING AND PACKING'.

ASK ( 'The delivery terms will Free on Board (FOB).
A.  Origin
B.  Destination at a remote site
C.  Destination not at a remote site
D.  No equipment will be produced and shipped
E.  Unknown',
Delivery_Terms_FOB,[ A, B, C, D, E]).

IF ?Delivery_Terms_FOB is A
THEN Include_Paragraph(?area,'Delivery_Terms_FOB-A') AND
     do(Levels_Of_Packing).

IF ?Delivery_Terms_FOB is B
THEN Include_Paragraph(?area,'Delivery_Terms_FOB-B') AND
     do(Levels_Of_Packing).
```

```
IF ?Delivery_Terms_FOB is C
THEN Include_Paragraph(?area,'Delivery_Terms_FOB-C') AND
     do(Levels_Of_Packing).

IF ?Delivery_Terms_FOB is D
THEN Include_Paragraph(?area,'Delivery_Terms_FOB-D') and
     do(TP_Delivery_Terms_FOB).

IF ?Delivery_Terms_FOB is E
THEN Do(Levels_Of_Packing).
End. (* Delivery_Terms_FOB *)

TOPIC Levels_Of_Packing.
ASK ( 'Will known levels of packaging and packing be specified
for all categories of equipment and materials to be delivered
under the contract?', Levels_of_Packing,[ YES, NO, UNKNOWN]).

IF ?Levels_Of_Packing is YES
THEN Include_Paragraph(?area,'Levels_Of_Packing-YES').
Do(Pack_Haz_Materials).
End. (* Levels_Of_Packing *)

TOPIC Pack_Haz_Materials.
ASK  ('Will the contractor prepare the packaging, packing,
marking, and shipment of hazardous, dangerous materials?',
Pack_Haz_Materials,[ YES, NO, UNKNOWN]).

IF ?Pack_Haz_Materials is YES
THEN Include_Paragraph(?area,'Pack_Haz_Materials-YES').
Do(Special_Containers).
End. (* Pack_Haz_Materials *)

TOPIC Special_Containers.
ASK ( 'Will specialized containers be required to meet
contractual requirements?', Special_Containers,[ YES, NO,
UNKNOWN]).

IF ?Special_Containers is YES
THEN Include_Paragraph(?area,'Special_Containers-YES').
do(Initial_Parts).
End. (* Special_Containers *)

TOPIC Initial_Parts.
ASK ( 'Will initial operational spare/repair parts be
required?', Initial_Parts,[ YES, NO, UNKNOWN]).

IF ?Initial_Parts is YES
THEN Include_Paragraph(?area,'Initial_Parts-YES').
Do(TP_Delivery_Terms_FOB).
End. (*Initial_Parts *)
```

```
TOPIC TP_Delivery_Terms_FOB.
write([?outfile],'TRANSPORTATION:',#n).
area is 'TRANSPORTATION'.

IF ?Delivery_Terms_FOB is A
THEN Include_Paragraph(?area,'TP_Delivery_Terms_FOB-A').

IF ?Delivery_Terms_FOB is B
THEN Include_Paragraph(?area,'TP_Delivery_Terms_FOB-B').

IF ?Delivery_Terms_FOB is C
THEN Include_Paragraph(?area,'TP_Delivery_Terms_FOB-C').

IF ?Delivery_Terms_FOB is_not D
THEN do(Security).
End. (* TP_Delivery_Terms_FOB *)

TOPIC Security.
ASK ( ' Will the system generate, handle, process, or use
classified information?', Security,[ YES, NO, UNKNOWN]).

IF ?Security is YES
THEN Include_Paragraph(?area,'Security-YES').
DO(Transport_Parameters).
End. (* Security *)

TOPIC Transport_Parameters.
ASK ( 'Does the equipment exceed the transportability
parameters listed in MIL-P-9024?', Transport_Parameters,[ YES,
NO, UNKNOWN]).

IF ?Transport_Parameters is YES
THEN Include_Paragraph(?area,'Transport_Parameters-YES').
End. (* Transport_Parameters *)

Topic Include_Paragraph(area,findstring).
Text is find('special.txt',,,?findstring,'/end').
window(?area,,,,1,7,78,13).
say(?text).
write([?outfile],?text,#n).
close_window().
close('special.txt').
end. (* Include_Paragraph *)

End. (* Specialty *)
```

```
(*                          SUPEQUIP.KB                          *)
(* SUPEQUIP.KB PRODUCES THE TEXT FOR PARAGRAPH -                 *)
(* 3.8 COMMON SUPPORT EQUIPMENT                                  *)

ASK('Input a file name',outfile).
do('Support Equipment').

(* 'SUPPORT EQUIPMENT' IS THE PARENT TOPIC                       *)
Topic 'Support Equipment'.
write([?outfile],'3.8 Common Support Equipment',#n).
area is 'COMMON SUPPORT EQUIPMENT'.
do(Restore).

Topic Restore.
EOF is number_to_char(26).
data is find('design.dat').
IF ?data <> ?EOF
THEN Type_Equipment is read('design.dat','Type_Equipment','/')
ELSE Type_Equipment is UNKNOWN.
do(Support_Equipment).
end. (* Restore *)

Topic Include_Paragraph(area,findstring).
Text is find('Supequip.txt',,,?findstring,'/end').
window(?area,,,,1,7,78,13).
say(?text).
write([?outfile],?text,#n).
close_window().
close('Supequip.txt').
end. (* Include_Paragraph *)

Topic Support_Equipment.
IF ?Type_Equipment is Simulators OR
   ?Type_Equipment is 'No Additional Equipment'
THEN Support_Equipment is D
ELSE
ASK('Support equipment will:
A. BE REQUIRED TO SUPPORT THE SYSTEM ORGANICALLY.
B. BE REQUIRED BY THE CONTRACTOR
C. NOT BE REQUIRED
D. UNKNOWN',
Support_Equipment, [A,B,C,D]).

IF ?Support_Equipment is A
THEN Include_Paragraph(?area,'Support_Equipment-A') and
     do(Support_Equipment_Plan).

IF ?Support_Equipment is B
THEN Include_Paragraph(?area,'Support_Equipment-B') and
     do(SE_Identification).
```

```
IF ?Support_Equipment is D
THEN do(Support_Equipment_Plan).

end. (* Support_Equipment_Plan *)

Topic Support_Equipment_Plan.
ASK('The Support Equipment Plan (SEP):',
Support_Equipment_Plan,['Is Required', 'Is Not Required',
UNKNOWN]).

IF ?Support_Equipment_Plan is 'Is Required'
THEN
Include_Paragraph(?area,'Support_Equipment_Plan-Is_Required').
do(SE_Identification).
end. (* Support_Equipment_Plan *)

Topic SE_Identification.
ASK('The type of Systems Engineering methodology used for
support equipment (SE) identification is:', SE_Identification,
[LSA, 'Informal Analysis', UNKNOWN]).

IF ?SE_Identification is LSA
THEN Include_Paragraph(?area,'SE_Identification-LSA').

IF ?SE_Identification is 'Informal Analysis'
THEN Include_Paragraph(?area,
'SE_Identification-Informal_Analysis').
end. (* SE_Identification *)

End. (* Support Equipment *)
```

```
(*                          SUPPORT.KB                          *)
(* SUPPORT.KB PRODUCES THE TEXT FOR SOW PARAGRAPH -  *)
(* TEST AND EVALUATION SUPPORT WHICH IS PART OF SOW *)
(* PARAGRAPH - 3.4 TEST AND EVALUATION                *)

ASK('INPUT FILE NAME',outfile).
do(Support).

(* TOPIC SUPPORT IS THE PARENT TOPIC      *)
Topic Support.
write([?outfile],'Test and Evaluation Support',#n).
do(Delete_Data).

Topic Delete_Data.
(* Delete old data *)
EOF is number_to_char(26).
data is find('support.dat').
IF ?data <> ?EOF
THEN dos('del support.dat') and
     new_file('support.dat') and
     close('support.dat').
do(Maintenance_Concept).
end.(*Delete_Data*)

Topic Include_Paragraph(area,findstring).
Text is find('support.txt',,,?findstring,'/end').
window(?area,,,,1,7,78,13).
say(?text).
write([?outfile],?text,#n).
close_window().
close('support.txt').
end. (* Include_Paragraph *)

Topic Showtext(area,findstring).
Text is find('support.txt',,,?findstring,'/end').
window(?area,,,,1,7,78,13).
say('NOTE: THE FOLLOWING MESSAGE IS NOT WRITTEN TO A
FILE.',#n,?text).
close_window().
close('support.txt').
end. (* Showtext *)
(*   Preoperational Maintenance   *)

Topic Maintenance_Concept.
write([?outfile],'PREOPERATIONAL MAINTENANCE:',#n).
area is 'PREOPERATIONAL MAINTENANCE'.

ASK('The Maintenance Concept:

A. NEEDS TO BE WRITTEN
B. IS WRITTEN, NEEDS TO BE UPDATED.
```

```
C. IS WRITTEN, DOES NOT NEED TO BE UPDATED
D. IS NOT REQUIRED
E. UNKNOWN',

Maintenance_Concept, [A, B, C, D, E]).

If ?Maintenance_Concept is A
THEN Include_Paragraph(?area,'Maintenance_Concept-A') and
     do(LSA_Task_302).

IF ?Maintenance_Concept is B
THEN Include_Paragraph(?area,'Maintenance_Concept-B') and
     do(LSA_Task_302).

IF ?Maintenance_Concept is C
THEN do(LSA_Task_302).

IF ?Maintenance_Concept is D
THEN do(GFE_Maintenance).

IF ?Maintenance_Concept is E
THEN do(LSA_Task_302).
end. (* Maintenance_Concept *)

Topic LSA_Task_302.
ASK('Has LSA Task 302, Support System Alternatives, been
tailored and accomplished, and have the results been included
in the ILSP/Maintenance Concept?', LSA_Task_302, [YES, NO,
UNKNOWN]).

IF ?LSA_Task_302 is NO
THEN Include_Paragraph(?area,'LSA_Task_302-NO').

IF ?LSA_Task_302 is UNKNOWN
THEN Include_Paragraph(?area,'LSA_Task_302-UNKNOWN').
do(GFE_Maintenance).
end. (* LSA_Task_302 *)

Topic GFE_Maintenance.
ASK('The contractor will perform preoperational maintenance:

A. ON GOVERNMENT FURNISHED EQUIPMENT
B. NOT ON GOVERNMENT FURNISHED EQUIPMENT
C. WILL NOT PERFORM PREOPERATIONAL MAINTENANCE
D. UNKNOWN.',
GFE_Maintenance, [A,B,C,D]).

IF ?GFE_Maintenance is A
THEN Include_Paragraph(?area,'GFE_Maintenance-A').

IF ?GFE_Maintenance is B
```

```
THEN Include_Paragraph(?area,'GFE_Maintenance-B').

IF ?GFE_Maintenance is D
THEN Include_Paragraph(?area,'GFE_Maintenance-D').

close('support.dat').
WRITE('Support.dat',#o,'GFE_Maintenance',
?GFE_Maintenance,'/').
do(Interim_Support).
end. (* GFE_Maintenance *)

Topic Interim_Support.
ASK('Will there be a requirement for interim contractor
support?', Interim_Support, [YES, NO, UNKNOWN]).

IF ?Interim_Support is YES
THEN Include_Paragraph(?area,'Interim_Support-YES').

IF ?Interim_Support is UNKNOWN
THEN Include_Paragraph(?area,'Interim_Support-UNKNOWN').
do(CETS_Required).
end. (* Interim_Support *)

Topic CETS_Required.
ASK('Is a Contractor Engineering and Technical Services (CETS)
Plan required?', CETS_Required, [YES, NO, UNKNOWN]).

IF ?CETS_Required is YES
THEN Include_Paragraph(?area,'CETS_Required-YES').

IF ?CETS_Required is UNKNOWN
THEN Include_Paragraph(?area,'CETS_Required-UNKNOWN').
do(Reliability_In_CDRL).
end. (* CETS_Required *)

Topic Reliability_In_CDRL.
ASK('Have reliability, maintainability data reporting and
feedback failure summary reports been included in the
reliability/maintainability section of the CDRL?',
Reliability_In_CDRL, [YES, NO, UNKNOWN]).

IF ?Reliability_In_CDRL is NO
THEN Include_Paragraph(?area,'Reliability_In_CDRL-NO').
do(Material_In_CDRL).
end. (* Reliability_In_CDRL *)

Topic Material_In_CDRL.
ASK('Have contractual provisions for material reporting been
included in the CDRL?', Material_In_CDRL, [YES, NO, UNKNOWN]).

IF ?Material_In_CDRL is NO
```

```
THEN Include_Paragraph(?area,'Material_In_CDRL-NO').
do(Preoperational_Spares_Required).
end. (* Material_In_CDRL *)

(*  Preoperational Supply Support *)

Topic Preoperational_Spares_Required.
write([?outfile],'PREOPERATIONAL SUPPLY SUPPORT:',#n).
area is 'PREOPERATIONAL SUPPLY SUPPORT'.

ASK('Will the contractor be required to provide spare and
repair parts for the equipment during the preoperational
phase?', Preoperational_Spares_Required, [YES, NO, UNKNOWN]).

IF ?Preoperational_Spares_Required is YES
THEN
Include_Paragraph(?area,'Preoperational_Spares_Required-YES')
and do(GFP_Spares).

IF ?Preoperational_Spares_Required is UNKNOWN
THEN Showtext(?area,'Preoperational_Spares_Required-UNKNOWN')
and  do(GFP_Spares).
end. (* Preoperational_Spares_Required *)

Topic GFP_Spares.
ASK('The equipment will include:

A. GFP WITH CONTRACTOR SPARE PARTS SUPPORT
B. GFP WITHOUT CONTRACTOR SPARE PARTS SUPPORT
C. NO GFP
D. UNKNOWN',
GFP_Spares, [A,B,C,D]).

IF ?GFP_Spares is A
THEN Include_Paragraph(?area,'GFP_Spares-A').

IF ?GFP_Spares is B
THEN Include_Paragraph(?area,'GFP_Spares-B').

IF ?GFP_Spares is D
THEN Showtext(?area,'GFP_Spares-UNKNOWN').
end. (* GFP_Spares *)

End. (* Support *)
```

```
(*                    TESTEVAL.KB                          *)
(* TESTEVAL.KB PRODUCES THE TEXT FOR SOW PARAGRAPH -  *)
(* 3.4 SYSTEM TEST AND EVALUATION                     *)

ASK('input a file name',outfile).
do('Test and Evaluation').

(* 'TEST AND EVALUATION' IS THE PARENT TOPIC          *)
Topic 'Test and Evaluation'.
write([?outfile],'3.4 System Test and Evaluation',#n).
area is 'SYSTEM TEST AND EVALUATION'.
do(Delete_Data).

Topic Delete_Data.
(* Delete old data *)
EOF is number_to_char(26).
data is find('testeval.dat').
IF ?data <> ?EOF
THEN dos('del testeval.dat') and
     new_file('testeval.dat') and
     close('testeval.dat').
do(Restore).
end.(* Delete_Data *)

Topic Restore.
EOF is number_to_char(26).
data is find('Design.dat').
IF ?data <> ?EOF
THEN Design_Change is read('Design.dat','Design_Change','/')
ELSE Design_Change is UNKNOWN.
do(Special_Oversight).
end. (* Restore *)

Topic Include_Paragraph(area,findstring).
Text is find('testeval.txt',,,?findstring,'/end').
window(?area,,,,1,7,78,13).
say(?text).
write([?outfile],?text,#n).
close_window().
close('testeval').
end. (* Include_Paragraph *)

Topic Showtext(area,findstring).
Text is find('testeval.txt',,,?findstring,'/end').
window(?area,,,,1,7,78,13).
say('NOTE: THE FOLLOWING MESSAGE IS NOT WRITTEN TO DISK',
?TEXT).
close_window().
close('testeval.txt').
end. (* Showtext *)
```

```
Topic Special_Oversight.
ASK ('Is your program a major program as defined in AFR 800-2
or
has it been designated for special oversight by the Director
of
Operational Test and Evaluation (DOT&E)?', Special_Oversight,
[YES, NO, UNKNOWN]).

IF ?Special_Oversight is YES
THEN Include_Paragraph(?area,'Special_Oversight-YES').

IF ?Special_Oversight is UNKNOWN
THEN Showtext(?area,'Special_Oversight-UNKNOWN').
do(Type_Testing_Required).
exit().
end. (* Special Oversight *)

Topic Type_Testing_Required.
ASK ('The contractor will be required to:

A. TO CONDUCT A CI/SUB-SYSTEM/SYSTEM LEVEL QUALIFICATION TEST
   PROGRAM (DEVELOPMENT TEST AND EVALUATION (DT&E) TEST
   PROGRAM.
B. TO PARTICIPATE IN OR SUPPORT SYSTEM INITIAL OPERATION TEST
   AND EVALUATION (IOT&E) TESTING
C. BOTH OF THE ABOVE
D. WILL NOT BE REQUIRED TO CONDUCT TESTING
E. UNKNOWN',
Type_Testing_Required, [A,B,C,D,E]).

IF ?Type_Testing_Required is A
THEN Include_Paragraph(?area,'Type_Testing_Required-A') and
     do(Install_And_Check).

IF ?Type_Testing_Required is B
THEN Include_Paragraph(?area,'Type_Testing_Required-B') and
     do(Install_And_Check).

IF ?Type_Testing_Required is C
THEN Include_Paragraph(?area,'Type_Testing_Required-A') and
     Include_Paragraph(?area,'Type_Testing_Required-B') and
     do(Install_And_Check).

IF ?Type_Testing_Required is D
THEN Include_Paragraph(?area,'Type_Testing_Required-D_or_E')
and  exit().

IF ?Type_Testing_Required is E
THEN Include_Paragraph(?area,'Type_Testing_Required-D_or_E')
and do(Install_And_Check).
```

```
end.(* Type_Testing_Required*)

Topic Install_And_Check.
IF ?Type_Testing_Required is D
THEN exit()
ELSE
ASK('Will the contactor be tasked to install and check on the
system (or equipment) at a fixed or permanent site prior to
conducting the system and/or IOT&E tests?', Install_And_Check,
[YES,NO,UNKNOWN]).

IF ?Install_And_Check is YES
THEN Include_Paragraph(?area,'Install_And_Check-YES').

IF ?Install_And_Check is UNKNOWN
THEN Showtext(?area,'Install_And_Check-UNKNOWN').

IF ?Type_Testing_Required is_not B
THEN do(FCA_OR_FQR).
end. (*Install_And_Check*)

Topic FCA_OR_FQR.
ASK('The contractor will be required to conduct a:',
FCA_OR_FQR, ['FUNTIONAL CONFIGURATION AUDIT (FCA)','FORMAL
QUALIFICATION REVIEW (FQR)','BOTH OF THE ABOVE', UNKNOWN]).

IF ?FCA_OR_FQR is 'FUNTIONAL CONFIGURATION AUDIT (FCA)' OR
   ?FCA_OR_FQR is 'FORMAL QUALIFICATION REVIEW (FQR)' OR
   ?FCA_OR_FQR is 'BOTH OF THE ABOVE'
THEN Include_Paragraph(?area,'FCA_OR_FQR-BOTH').

close('testeval.dat').
WRITE('testeval.dat',#o,'FCA_OR_FQR',?FCA_OR_FQR,'/').
do(Software_Development).
end.(*FCA_OR_FQR*)

Topic Software_Development.
IF ?Design_Change is Software or
   ?Design_Change is Both
THEN Software_Development is YES
ELSE
IF ?Type_Testing_Required is D
THEN exit()
ELSE
ASK('Will the contractor be required to perform software
development,modification, and/or integration as part of the
overall system development?',Software_Development,
[YES,NO,UNKNOWN]).

IF ?Software_Development is YES
THEN Include_Paragraph(?area,'Software_Development-YES').
```

```
IF ?Software_Development is UNKNOWN
THEN Showtext(?area,'Software_Development-UNKNOWN').
do(Be_In_TPWG).
end. (*Be_In_TPWG*)

Topic Be_In_TPWG.
ASK('Will the contractor be required to be a member of the
Test Planning Working Group (TPWG)?', Be_In_TPWG, [YES,NO,
UNKNOWN]).

IF ?Be_In_TPWG is YES
THEN Include_Paragraph(?area,'Be_In_TPWG-YES').

IF ?Be_In_TPWG is UNKNOWN
THEN Showtext(?area,'Be_In_TPWG-UNKNOWN').
do(Verification_Cross_Reference).
end. (*Be_In_TPWG*)

Topic Verification_Cross_Reference.
ASK('Will the contactor be required to prepare or complete the
verification cross reference index appearing in Section 4
(quality assurance) of the HWCI/CSCI specification(s)?',
Verification_Cross_Reference, [YES, NO,UNKNOWN]).

IF ?Verification_Cross_Reference is YES
THEN
Include_Paragraph(?area,'Verification_Cross_Reference-YES').

IF ?Verification_Cross_Reference is NO
THEN Showtext(?area,'Verification_Cross_Reference-NO').
do(Special_Facilities).
end. (*Verification_Cross_Reference*)

Topic Special_Facilities.
ASK('Will the contractor be required to arrange for the use of
special or unique commercial and/or government test
facilities?', Special_Facilities, [YES,NO,UNKNOWN]).

IF ?Special_Facilities is YES
THEN Include_Paragraph(?area,'Special_Facilities-YES').

IF ?Special_Facilities is UNKNOWN
THEN Showtext(?area,'Special_Facilities-UNKNOWN').
do(Radio_Frequency_Allocation).
end.(*Special_Facilities*)

Topic Radio_Frequency_Allocation.
ASK('Was a radio frequency allocation required and obtain for
contractor development for in-plant testing?  If so, will the
contractor be tasked to perform system testing or installation
```

```
and checkout at a new location not referenced in the original
application?', Radio_Frequency_Allocation, [YES, NO,
UNKNOWN]).

IF ?Radio_Frequency_Allocation is YES
THEN
Include_Paragraph(?area,'Radio_Frequency_Allocation-YES').

IF ?Radio_Frequency_Allocation is UNKNOWN
THEN Showtext(?area,'Radio_Frequency_Allocation-UNKNOWN').
do(Software_Test_Program).
end.(*Radio_Frequency_Allocation*)

Topic Software_Test_Program.
IF ?Design_change is Hardware
THEN Software_Test_Program is NO
ELSE
IF ?Type_Testing_Required is D
THEN exit()
ELSE
ASK('Will the contractor be conducting a formal computer
software test program?', Software_Test_Program, [YES,NO,
UNKNOWN]).

IF ?Software_Test_Program is YES
THEN Include_Paragraph(?area,'Software_Test_Program-YES').
close('testeval.dat').
WRITE('testeval.dat',#o,'Software_Test_Program',?Software_Test
_Program,'/').
do(Maintain_Test_Equipment).
end.(* Softwaare_Test_Program *)

Topic Maintain_Test_Equipment.
ASK('Will the contractor be required to maintain and calibrate
test equipment?', Maintain_Test_Equipment, [YES, NO,
UNKNOWN]).

IF ?Maintain_Test_Equipment is YES
THEN Include_Paragraph(?area,'Maintain_Test_Equipment-YES').
do(Test_Requirements).
end. (*Maintain_Test_Equipment*)

Topic Test_Requirements.
ASK('Will the contractor be required to prepare performance
diagnostic test data or test procedures for automatic,
semi-automatic or manual test equipment?', Test_Requirements,
[YES, NO, UNKNOWN]).

IF ?Test_Requirements is YES
THEN Include_Paragraph(?area,'Test_Requirements-YES').
do(Prod_Test_Procedures).
```

```
end. (*Test_Requirements*)

Topic Prod_Test_Procedures.
ASK('Will the contractor be required to prepare and submit
production test procedures?', Prod_Test_Procedures, [YES, NO,
UNKNOWN]).

IF ?Prod_Test_Procedures is YES
THEN Include_Paragraph(?area,'Prod_Test_Procedures-YES').
do(LRIP_Program).
end. (* Prod_Test_Procedures *)

Topic LRIP_Program.
ASK('Will the contactor be required to institute a Low Rate
Initial Production (LRIP) Program?', LRIP_Program, [YES, NO,
UNKNOWN]).

IF ?LRIP_Program is YES
THEN Include_Paragraph(?area,'LRIP_Program-YES').

IF ?LRIP_Program is UNKNOWN
THEN Showtext(?area,'LRIP_Program-UNKNOWN').
do(SR_System).
end. (*LRIP_Program*)

Topic SR_System.
ASK('Will the contractor be required to establish a Service
Reporting (SR) system?', SR_System, [YES, NO, UNKNOWN]).

IF ?SR_System is YES
THEN Include_Paragraph(?area,'SR_System-YES').
end. (*SR_System*)

end. (* Test and Evaluation *)
```

```
(*                          TRAINING.KB                                *)
(* TRAINING.KB PRODUCES THE TEXT FOR SOW PARAGRAPH -                   *)
(* 3.3 TRAINING                                                        *)

do('Training').

(* 'TRAINING' IS THE THE PARENT TOPIC                                  *)
Topic 'Training'.
write([?outfile],'3.2 Training',#n).
area is 'TRAINING'.
do(Restore).

Topic Restore.
EOF is number_to_char(26).
data is find('logistic.dat').
IF ?data <> ?EOF
THEN Warranty_Maintenance is
read('logistic.dat','Warranty_Maintenance','/')
ELSE Warranty_Maintenance is UNKNOWN.

data is find('Design.dat').
IF ?data <> ?EOF
THEN Type_Equipment is read('Design.dat','Type_Equipment','/')
ELSE Type_Equipment is UNKNOWN.

do(Training_Required).
end. (* Restore *)


Topic Include_Paragraph(area,findstring).
Text is find('Training.txt',,,?findstring,'/end').
window(?area,,,,1,7,78,13).
say(?text).
write([?outfile],?text,#n).
close_window().
close('Training.txt').
end. (* Include_Paragraph *)


Topic Training_Required.
IF ?Warranty_Maintenance is A OR
   ?Warranty_Maintenance is B
THEN Training_Required is YES
ELSE ASK('Will your system require training for operation and
maintenance?',
Training_Required, [YES, NO, UNKNOWN]).

IF ?Training_Required is YES
THEN Include_Paragraph(?area,'Training_Required-YES') and
     do(Training_As_CLIN).
```

```
IF ?Training_Required is NO
THEN Include_Paragraph(?area,'Training_Required-NO').

IF ?Training_Required is UNKNOWN
THEN do(Training_As_CLIN).
end. (* Training_Required *)

Topic Training_As_CLIN.
ASK(' Will training be included as a contract line item in the
FSD contract?

NOTE: Normally, training requirements are procured solely by
ATC; Therefore, you should normally provide a NO response to
this question!', Training_As_CLIN, [YES, NO, UNKNOWN]).

IF ?Training_As_CLIN is YES
THEN Include_Paragraph(?area,'Training_As_CLIN-YES').

IF ?Training_As_CLIN is NO
THEN Include_Paragraph(?area,'Training_As_CLIN-NO').
do(Training_Equipment).
End.  (* Training_As_CLIN *)

Topic Training_Equipment.
IF ?Type_Equipment is Simulators OR
   ?Type_Equipment is Both
THEN Training_Equipment is YES
ELSE ASK('Will your system include the development of training
equipment?

NOTE:  Normally, training equipment is not developed during
this phase. You should normally provide a NO response to this
question!', Training_Equipment, [YES, NO, UNKNOWN]).

IF ?Training_Equipment is YES
THEN Include_Paragraph(?area,'Training_Equipment-YES').

IF ?Training_Equipment is NO
THEN Include_Paragraph (?area,'Training_Equipment-NO').

do(Contingency_Plan).
end. (*Training_Equipment *)

Topic Contingency_Plan.
ASK('If Technical Orders/Manuals are not available in time for
training, is there a contingency plan?', Contingency_Plan,
[YES, NO, UNKNOWN]).

IF ?Contingency_Plan is YES
THEN Include_Paragraph(?area,'Contingency_Plan-YES').
```

```
IF ?Contingency_Plan is NO
THEN Include_Paragraph(?area,'Contingency_Plan-NO').
end. (* Contigency_Plan *)

end. (* Training *)
```

```
REM:                        KP.BAT
REM: THIS IS THE BATCH FILE WHICH AUTOMATICALLY EXECUTES
REM: KNOWLEDGEPRO.
REM: NOTE THAT THIS REVISION OF KP.BAT ALLOWS THE USE OF THE
REM: FIND COMMAND.
echo off
set _CHAIN=
set _USER=97
set _PRM=
find
if exist kp0.com goto kp
if exist dkp0.com goto dkp
rkp0 %1 %2
goto end
:kp
kp0 %1 %2
goto end
:dkp
dkp0 %1 %2
goto end
:end
if errorlevel 127 %_chain%
set _USER=
set _CHAIN=
set _PRM=
```

Appendix E: Topic Reference Chart

Topic	Software Module
Access_Roads	Facility.kb
Alternative_Support_Concepts	Logistic.kb
Analysis_Required	Special.kb
Analyze_ECPS	Special.kb
Availability	Logistic.kb
BCS_Required	Logistic.kb
Be_In_TPWG	Testeval.kb
Be_In_Weather	Special.kb
CETS_Required	Support.kb
Close_Loop_Track	Special.kb
Communications	Commun.kb
Computer_Resources	Pmgmt.kb
Config_audits	Design.kb
Config_Required	Pmgmt.kb
Contigency_Plan	Training.kb
Contract	Logistic.kb
Contractor_as_Integrator	Special.kb
Contractor_Config	Pmgmt.kb
Contractor_Install	Security.kb
Contract_Type	Pmgmt.kb
Count_Config_Items	Pmgmt.kb
Data	Data.kb
Data_Management	Data.kb
Delete_Data	Design.kb
Delete_Data	Logistic.kb
Delete_data	Menu.kb
Delete_Data	Pmgmt.kb
Delete_Data	Support.kb
Delete_Data	Testeval.kb
Delivery_Terms_FOB	Special.kb
Design	Pmgmt.kb
Design_Change	Design.kb
Design_Parameters	Logistic.kb
Dev_Contractor_Is_Prod	Pmgmt.kb
DOLLARS	Design.kb
Dollars_FSD_Contract	Pmgmt.kb
Dollars_Prod_Contract	Pmgmt.kb
ECPS	Pmgmt.kb
Electromagnetic_Radiation	Special.kb
Electromag_Radiation	Commun.kb
Engineering_Data_Required	Data.kb
Environmental_Deliverables	Facility.kb
Equipment_Location	Design.kb
Facilities	Facility.kb

Facilities_Task_Required	Facility.kb
FCA_OR_FQR	Testeval.kb
Formal_Qual_Review	Pmgmt.kb
FQR	Design.kb
Funding_Cap	Pmgmt.kb
GFE_Maintenance	Support.kb
GFE_OR_Utilities	Facility.kb
GFE_Required	Special.kb
GFP_Spares	Support.kb
Hardware	Data.kb
Hardware_Supported_By	Spares.kb
Hazardous_Materials	Logistic.kb
Helptext	Menu_Sys.kb
Help_Management	Menu_Sys.kb
Help_System	Menu_Sys.kb
How_Big	Facility.kb
How_Travel	Pmgmt.kb
Human_Factors	Design.kb
ILS_Required	Logistic.kb
Impact	Logistic.kb
Impact_Assessment	Logistic.kb
Include_Paragraph	Commun.kb
Include_Paragraph	Data.kb
Include_Paragraph	Design.kb
Include_Paragraph	Facility.kb
Include_Paragraph	Logistic.kb
Include_Paragraph	Pmgmt.kb
Include_Paragraph	Security.kb
Include_Paragraph	Spares.kb
Include_Paragraph	Special.kb
Include_Paragraph	Supequip.kb
Include_Paragraph	Support.kb
Include_Paragraph	Testeval.kb
Include_Paragraph	Training.kb
Initial_Parts	Special.kb
Initial Spare/Repair Parts	Spares.kb
Install_And_Check	Testeval.kb
Interface_Control	Pmgmt.kb
Interim_Support	Support.kb
Joint_Service_Program	Logistic.kb
Lessons_Learned	Special.kb
Levels_Of_Packing	Special.kb
Logistics_Engineering	Logistic.kb
Long_Line_Communications	Commun.kb
LRIP_Program	Testeval.kb
LSA_Task_302	Support.kb
Maintainability_Required	Logistic.kb
Maintain_Test_Equipment	Testeval.kb
Maintenance_Concept	Support.kb
Main_Menu	Menu.kb
Material_In_CDRL	Support.kb

Maximize_Resources	Logistic.kb
Mishap_Risk	Special.kb
NDI	Special.kb
NDI_ITEMS	Logistic.kb
NDI_USED	Pmgmt.kb
New_Tech_Impact	Logistic.kb
Nomen_Required	Pmgmt.kb
O&M_Required	Facility.kb
Pack_Haz_Materials	Logistic.kb
Pack_Haz_Materials	Special.kb
Parts_Control	Design.kb
Periodic_Reports	Special.kb
Position_Description	Design.kb
Prelim_Hazard	Special.kb
Preopertional_Spares_Required	Support.kb
Prime_Mission_Equip	Commun.kb
Prime_Mission_Equipment	Design.kb
Prod_Test_Procedures	Testeval.kb
Program Management	Pmgmt.kb
Provisioning_Guidance_Conference	Spares.kb
Radio_Frequency_Allocation	Testeval.kb
Real Property Facilities	Facility.kb
Reliability_In_CDRL	Support.kb
Reliability_Parameters	Logistic.kb
Reliability_Required	Logistic.kb
Repair_Level_Analysis	Logistic.kb
Restore	Commun.kb
Restore	Data.kb
Restore	Facility.kb
Restore	Pmgmt.kb
Restore	Special.kb
Restore	Supequip.kb
Restore	Testeval.kb
Restore	Training.kb
Safety_At_Reviews	Special.kb
Safety_Safe_Design	Special.kb
Safety_Test	Special.kb
Safety_Training	Special.kb
SAIP	Spares.kb
SDR	Design.kb
Secure_Comm_Required	Security.kb
Security	Security.kb
Security	Special.kb
Security_Required	Security.kb
Security_Thru_Documentation	Security.kb
SEMP_Submitted	Design.kb
SE_Identification	Supequip.kb
Showtext	Design.kb
Showtext	Facility.kb
Showtext	Support.kb
Showtext	Testeval.kb

Site_Selected	Facility.kb
Software_Analysis	Special.kb
Software_Development	Testeval.kb
Software_Test_Program	Testeval.kb
Specialty	Special.kb
Special_Containers	Special.kb
Special_Facilities	Testeval.kb
Special_Oversight	Testeval.kb
Special_Requirements	Logistic.kb
SR_System	Testeval.kb
SSG_Or_SSWG	Special.kb
SSPP_Written_By	Special.kb
Status_Acct_Reports	Pmgmt.kb
Support	Support.kb
Support_Design_Constraints_Doc	Logistic.kb
Support_Equipment	Supequip.kb
Support_Equipment	Supequip.kb
Support_Equipment_Plan	Supequip.kb
Support_Factors_Documentation	Logistic.kb
sv_program_plan	Design.kb
sv_required	Design.kb
System/Project Management Menu	Menu_Sys.kb
Systems Engineering	Design.kb
System_assembled	Design.kb
System Designed or Developed	Pmgmt.kb
System_Des_Or_Mod	Pmgmt.kb
System Management Menu	Menu_Sys.kb
System_Safety_Manager	Special.kb
Tasks_For_Operation	Logistic.kb
Tasks_For_Source_Data	Logistic.kb
Technical_Orders	Data.kb
Test And Evaluation	Testeval.kb
TEST_EVAL	Design.kb
Test_Requirements	Testeval.kb
Towers_Required	Facility.kb
To_Deliverable	Data.kb
TO_Government_Ownership	Facility.kb
TP_Delivery_Terms_FOB	Special.kb
Training	Training.kb
Training_As_CLIN	Training.kb
Training_Equipment	Training.kb
Training_Required	Training.kb
Transport_Parameters	Special.kb
Travel_To	Pmgmt.kb
Type_Baseline	Pmgmt.kb
Type_Comsec_Material	Security.kb
Type_Equipment	Design.kb
TYPE_FSD_EFFORT	Logistic.kb
Type_Prod_Contract	Pmgmt.kb
Type_Testing_Required	Testeval.kb
Unique_Sched_Management	Pmgmt.kb

Use_Weather_Data	Special.kb
Verification_Cross_Reference	Testeval.kb
Warranty_Maintenance	Logistic.kb
Who_Tempest_Test	Security.kb

Appendix F: Definitions

Artificial Intelligence - The part of computer science which deals with intelligent computer programs. (3:5)

Algorithms - Formulas which produce a correct answer when numerical data is inserted into the formula. (14:29)

Experts - A person who uses their experience and knowledge to solve certain types of problems. (11:1)

Expert Systems - "An intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solution [3:5]."

Heuristics - A rule of thumb and often called the art of good quessing. Heuristics enable "experts to break problems down to smaller components, to get around incomplete data, and to make educated guesses when necessary [16:74]."

Hypertext - A database management system which accesses data by reference. It searches for a string of text in a data base and returns any information associated with that string of text.

Inference - The process of deriving new facts from known facts. (4:261)

Knowledge Base - The part of the expert system containing the experts knowledge about a particular task. (5:23)

Inference Engine - The part of the expert system which contains the rules to solve a problem. (18:116)

Statement of Work (SOW) - A document which communicates the tasks and work effort the government wants the contractor to perform.

Work Breakdown Structure (WBS) - A concept scheme which breaks down a system into its components and subcomponents and work effort into its tasks and subtasks. (2:7)

Bibliography

1. David, James C. and Joseph S. Price Successful Communication in Full Scale Engineering Development Statements of Work. MS thesis, AFIT/LSSR-18-80. School of Systems and Logistics, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, June 1980 (ADA087497).

2. Department of Defense. Preparation of a Statement of Work (SOW). MIL-HDBK-245B. Washington: Government Printing Office, 1 June 1983.

3. Geissman, James R. and Roger D Schultz. "Verification and Validation of Expert Systems," AI Expert, 3: 26-33 (February 1988).

4. Harmon, Paul and David King. Expert Systems: Artificial Intelligence in Business. New York: John Wiley and Sons, 1985.

5. Hart, Anna. Knowledge Acquisition for Expert Systems. New York: McGraw-Hill Book Company, 1986.

6. Hayes-Roth, Fredrick and others. Building Expert Systems. Reading MA: Addison-Wesley Publishing Company, 1983.

7. Hazen, Christopher M. How Can Air Force Civil Engineers Use Expert Systems. MS thesis, AFIT/GEM/LSM/ 88S-9. School of Systems and Logistics, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, August 1988.

8. Headquarters Air Force Systems Command (USAF). Statement of Work (SOW). AFSCR 800-6. Washington: HQ AFSC, 3 April 1986.

9. Jackson, Peter. Introduction to Expert Systems. Reading MA: Addison-Wesley Publishing Company, 1986.

10. Klahr, Philip and Donald A. Waterman. Expert Systems: Techniques, Tools and Applications. Reading MA: Addison-Wesley Publishing Company, 1986.

11. Kulikowski, Casimir A. and Sholom M. Weiss. A Pratical Guide to Designing Expert Systems. Totowa NJ: Rowman and Allanheld, Publishers, 1984.

12. McCain, Steven A. An Expert System For Asset Reconciliation. MS thesis, AFIT/GLM/LSY/87S-46. School of Systems and Logistics, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, September 1987 (ADA186984).

13. Plumb, W. Airborne Systems Software Acquisition Engineering Guidebook for Statements of Work and Request For Proposal. Contract F3365776C0677. Redondo Beach CA: TRW Defense and Space Systems Group, September 1978 (ADA076544).

14. Prerau, David S. "Selection of an Appropriate Domain for an Expert System," The AI Magazine: 26-30 (Summer 1985).

15. Price, Patrick. Chief Systems Engineer. Personal Interview. ASD/YWSE, Wright-Patterson AFB OH, May 1989.

16. Sell, Peter S. Expert Systems - A Pratical Introduction. New York: John Wiley and Sons, 1985.

17. Shurkin, Joel N. "Expert Systems: The Practical Face of Artificial Intelligence," Technology Review: 72-78 (November/December 1983).

18. Van Horn, Mike. Understanding Expert Systems. New York: Bantam Books, 1986.

19 Waterman, Donald A. A Guide to Expert Systems. Reading MA: Addision-Wesley Publishing Company, 1986.

Vita

Captain Keith A. Dierking [redacted] He attended Rock Valley College from which he received an Associate of Science degree in Engineering in May 1980. He then transferred to the University of Illinois from which he received a Bachelor of Science degree in Computer Engineering in December 1982. In May 1983, he received a commision in the USAF from the Officer Training School (OTS). He then served as a Computer Systems Design Engineer at Training Systems System Program Office (SPO) at Wright-Patterson AFB, Ohio. In 1987, he became the F-15 OFT Lead Systems Engineer until entering the School of Systems and Logistics, Air Force Institute of Technology, in June 1988.

[redacted]

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for release; distribution unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) AFIT/GSM/LSM/89S-6	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION School of Systems and Logistics 6b. OFFICE SYMBOL (If applicable) AFIT/LSY	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) Air Force Institute of Technology Wright-Patterson AFB OH 45433-6583	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
AN EXPERT SYSTEM FOR DEVELOPING A FULL SCALE DEVELOPMENT STATEMENT OF WORK

12. PERSONAL AUTHOR(S)
Keith A. Dierking, B.S., Captain, USAF

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
MS Thesis	FROM _____ TO _____	1989 September	164

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Expert System, Statement of Work, Computer Applications, Artificial Intelligence, Acquisition, procurement
12	9		

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

Thesis Chairman: James R. Holt, Lt Col, USAF
Assistant Professor of Engineering Management

Approved for public release: IAW AFR 190-1.

LARRY W. EMMELHAINZ, Lt Col, USAF 14 Oct 89
Director of Research and Consultation
Air Force Institute of Technology (AU)
Wright-Patterson AFB OH 45433-6583

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	UNCLASSIFIED
22a NAME OF RESPONSIBLE INDIVIDUAL James R. Holt, Lt Col, USAF	22b. TELEPHONE (Include Area Code) (513)255-5023 / 22c. OFFICE SYMBOL AFIT/LSY

DD Form 1473, JUN 86 *Previous editions are obsolete.* SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

The purpose of this project is to determine the need and feasibility of developing an expert system to assist in the development of a Statement of Work (SOW) for the Full Scale Development (FSD) phase of the acquisition cycle. This project also determines the feasibility of transporting this expert system to a microcomputer.

The methodology involves a six step process for developing a small scale expert system. The first step involves choosing a tool. The tool chosen for this project is KnowledgePro by Knowledge Garden. The second step involves defining the problem. Once the problem is defined, a determination is made on whether an expert system is appropriate for this project using a model developed in a previous thesis project. The third step involves developing the system. The forth step involves developing a prototype system. The prototype system was developed on an IBM compatible computer using the KnowledgePro development system. The fifth step involved expanding, testing, and revising the system. The sixth step involves maintaining and updating the system. Besides the six steps for developing the expert system, the system was validated.

The project resulted in a microcomputer based expert system for assisting managers in the development of a FSD SOW. The system produces a document in the Work Breakdown Structure as called out by MIL-STD-245B consisting of essential SOW entries, potential documents to incorporate into the SOW, and action messages which provides direction in writing the SOW.

The benefit from this research is not limited to the development of the SOW. The methodology used in this research project can be used in the development of item specifications, contract development, Test and Evaluation Master Plans (TEMPs), acquisition plans, Program Management Plans (PMPs), and Contract Data Requirement Lists (CDRLs)